目　　录



目　录

前　言

本书是从《科学》杂志(Scientific American 中文版)自 1978 年创刊以来发表的数学游戏及计算机游戏专栏文章中精选近 100 例汇编而成的。《Scientific American》从 1952 年起开辟《数学游戏》专栏，特邀著名游戏数学专家马丁·加德纳及其他数学家与计算机科学家先后为该专栏撰稿，至今不断。该专栏的内容广涉数学游戏和计算机游戏的各种问题和各个领域，集知识性、趣味性和娱乐性为一体。作者以深入浅出地把一些复杂的数学和计算机问题通过常见的例子介绍给广大读者，行文浅显易懂，叙述生动活泼，没有深奥的公式和抽象的论证。凡具有高中数学水平的人一般都能看懂。

本书适合于广大中学生和大学生阅读；对于爱好数学的成人，本书也有相当大的吸引力。我们希望它能起到启迪思维、开阔视野、激发想象力和创造力的作用。本书内容如有错误或不当之处，敬请广大读者批评指正。

1. 金字塔的数学问题

古埃及的金字塔属于所有考古学之谜中最令人感到神秘莫测的问题之列。最大的金字塔——吉萨大金字塔——是埃及国王胡夫在公元前 2500 年左右建造的,至今仍基本上完好无损。

它最初的高度将近 147 米(482 英尺),重量超过 70 亿公斤(770 万吨)。这座金字塔是用巨大的石块建造的,石块从采石场上开采出来,修整成相当整齐的形状并运到建筑工地上,然后以令人瞠目的精确度一块块地堆叠起来。

埃及人是怎样建起这样庞大的建筑物的呢?为什么要修建金字塔?历史学家和考古学家对这些问题的了解并不是很有把握。许多金字塔是用作国王的坟墓,而关于金字塔的建造则有各种各样的说法。由于一位业余的古埃及研究者 Stuart Kirkland Wier 的某些探测性数学研究成果,现在历史学家能够较有把握地估计建造金字塔的民工队伍的规模。

在大金字塔建成后约 2000 千年,希腊历史学家 Herodotus 报道说,建造这座大金字塔动用了 10 万名民工。然而,Herodotus 的说法并不是关于古代埃及情况的可信资料。现在看来,他似乎把民工人



数高估了整整一个数量级。据 Wier 说，真正的人数少得出人意料：只有大约 1 万人。Wier 并不知道金字塔是如何建造的，那么他是怎样得出这个数字的呢？他首先估计出修建金字塔所需的工作量，然后对埃及人如何安排工程进度作出若干假定。

Wier 的计算是针对胡夫金字塔的，但这同一方法也适用于其他金字塔。第一步是要计算出一座金字塔蕴含了多少"能量"。在这种场合中，所谓能量指的是势能，即把给定质量提升到一定高度需要做的功。Wier 把金字塔的势能除以金字塔建造的时间(天数)，就得出了每天提升石块所需的工作量。古埃及的民工唯一可用的动力源就是人的肌肉。把需要做的工作量除以埃及民工平均的能量输出，Wier 就确定了完成提升工作所需的民工人数(见图 1)。

最大的一个不确定因素是时间。建造胡夫金字塔用了多少时间呢？胡夫在位的时间为 23 年。他的金字塔的建造工程大概不会在他登基以前就开始，但有可能在他去世前不久完工，也有可能在他去世前很多年完工。Wier 假定，建造胡夫金字塔所用的时间为 23 年——正好跟他在位的时间一样长。如果金字塔的施工是一年到头不间断地进行，那么这就相当于 8 400 天，实际的建造时间可能只有上述假定值的一半；由于存在这种根本的不确定因素，我们只能对民工数目作一个大略的估计。

胡夫金字塔在刚建成时有 146.7 米高，其正方形底座的边长为 230.4 米。高为 h，底边长为 S 的一座金字塔的体积为 $S^2 \times h/3$。对于胡夫大金字塔，根据这一公式可以算出它的体积为 260 万立方米。建造金字塔的材料是石灰石，其密度(d)为每立方米 2 700 公斤，因此该金字塔的质量达 70 亿公斤。金字塔的势能为 $h^2 \times d \times S^2 \times g/12$，其中 g 代表重力加速度(每秒每秒 9.81 米)。对于大金字塔，其势能为 2.52 万亿焦耳(焦耳是标准能量单位)。

民工每天所做的有用功平均为 24 万焦耳。假定效率为百分之



百,那么在8 400天内把所有石块安放到位将需要1 250名民工。这一估计量显然偏低。考虑到人力效率的低下,Wier把这一估计值乘以一个1.5的因数。他同时也考虑了把石块从胡夫石场的底部提升到金字塔工地所需的工作量(这一高度已知为19米)。

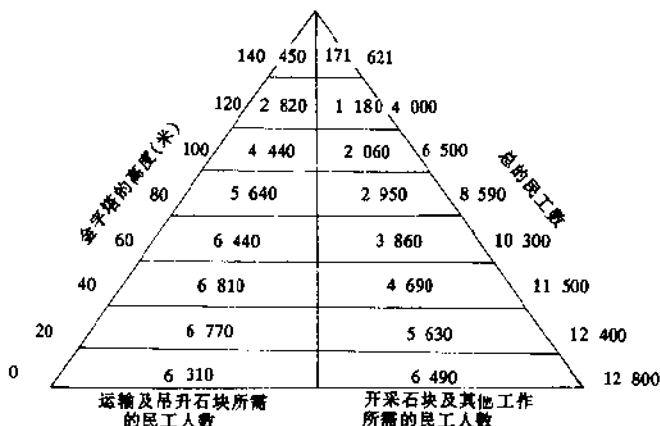


图1　修建金字塔所需的劳动力

当然,金字塔并不只是一堆没有任何结构的石头。金字塔内有通道和墓室,其中有些本身就是了不起的工程杰作。但是金字塔工程主要是把石块堆叠起来,因此我们可以忽略结构细节。金字塔是从底到顶一层层建造起来的。肯定只有在下面一层石块已经安放好后,才能继续增加石块构成新的一层。没有人知道石块是如何提升上去的,有些专家认为,埃及人是沿着沙堆成的巨大坡道把石块运上去的。其他专家则说民工们使用了巧妙的杠杆装置。

为了把石块从采石场运到建筑工地,民工们把石块放在木橇上拖运(古埃及的雕刻显示了民工拖石头的情景)。胡夫采石场距金字塔的底部有几百米远。为了计算出拖运石头所需的民工数,Wier估计出装了石块的木橇与地面的摩擦力,然后确定了克服摩擦力所需做的功。其他一些工作也需要有人去做:开采石块,把石块修整成形



以及把石块在金字塔中安装好等等。Wier 假定，所有这些工作所需要的劳动力最多为每立方米石块每天 14 位民工。

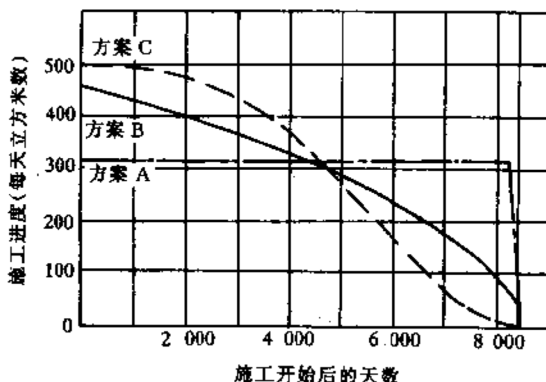


图 2　按照 3 种可能的施工方案，大金字塔都在 8 400 天内建成

这些计算得出了吉萨金字塔工程民工的平均数目。但是负责修金字塔的人是否始终都使用一样多的民工？还是在工程需要时征用更多的民工，而在不需要他们时又把他们打发走？有关资料没有记载这方面的情况，但可以推想存在几种可能性。

把大金字塔的体积除以建造它所需的时间，就可以知道平均每天必须把 310 立方米的石块安装好。然而，随着金字塔不断升高，石块提升的高度也越来越高，因此提升石块的工作量也越来越大。而随着金字塔的升高，民工们可以活动的工作场所——即未完工的金字塔的顶部面积——则越来越小。考虑到这些因素，始终保持每天 310 立方米的施工进度是不切实际的，就很清楚了。当金字塔较低时，施工进度应当比较快，并随着金字塔高度的增加而逐步下降。

Wier 考虑了 3 种可能的施工方案(见图 2)。方案 A 要求民工们在第 8 110 天以前以每天 315 立方米的不变进度安放石块。到第 8 110 天时，由于金字塔顶部的地方已经不大，施工进度将急剧下



降。施工方案B要求施工进度开始时为每天462立方米，此后逐步下降，到第8 000天后下降得更快。而方案C则要求施工进度开始时为每天500立方米，但在第2 000天后开始迅速下降，而在第7 000天后再逐渐下降到零，所有这3个方案都使得金字塔在8 400天中完工，但是随着工程的进展它们需要的民工人数是不同的。

图1的右下部分表示，根据方案B，在金字塔施工的每一阶段所需要的民工人数。施工开始时约有一半的民工在从事运送和提升石块的工作，另一半民工则在开采和安放石块以及从事其他各种工作。不过，到工程快要结束时，有3/4的民工在运送石块。在施工的任何阶段上，民工的人数都不超过12 800人——约为当时埃及估计人口总数的1%。方案A和方案C也得到类似的结果。

或许最简单的方案是使民工队伍始终保持恒定的规模，除非在工程即将结束的时候，此时金字塔的顶端大概只能容纳几个人了。按照这一方案，只需10 600名民工便足以建造胡夫的大金字塔。建造吉萨和达苏尔的其他那些大金字塔所需的民工人数就更少了。尽管这些陵墓的规模非常宏大，但古埃及还是有足够的民工——可以说绰绰有余——来建造它们。

2. 放弃平均律

假定我在不停地掷一枚均匀的硬币并连续不断地统计硬币的正面向上和反面向上的次数(所谓均匀的硬币是指它的正面向上和反面向上的可能性一样大,即各有1/2的概率)。如果到某个时候我掷出的正面向上的次数比反面向上的次数多100次,那么当我继续掷下去时,是否可能会出现一种反面向上的次数“赶上”正面向上次数的趋势呢?有些人根据在掷一枚均匀的硬币时正反两面出现次数最终应该相当这样一种直觉而认为平均律应当起作用。另外一些人则声称,硬币没有“记忆”——这样正面向上或反面向上的概率都始终各保持为1/2——并因而推论说根本不会出现两种次数扯平的趋势。

同样的问题也会出现在其他各种场合中。如果飞机坠毁事故平均每4个月发生一次,而已经有3个月没有发生坠机事故了,那么你是否预计很快将发生一起坠机事故?

在所有这些情况中,答案都是否定的。有关的随机过程——更确切地说是这些随机过程的标准数学模型——的确是没有任何记忆



的。

然而，这个问题在很大程度上与你说的“赶上”是什么意思有关。一连许多次掷出正面向上这一情况并不会影响以后掷币过程中出现反面的概率。即使如此，在正面向上的次数已经比反面向上的次数多了比如说 100 次以后，到某个阶段时两种次数将再次扯平的概率依然是 1。说某件事的概率为 1 通常意味着这件事是肯定无疑的，而概率为 0 则意味着是不可能出现的(在这个例子中，我们研究的是可能为无穷多次的掷硬币，因此数学家们喜欢用“几乎肯定会出现”和“几乎不可能出现”这种说法)。

我得马上又补充一句：在某种意义上也可以说掷硬币并不存在从长远来看两种结果扯平的趋势。例如，在正面向上的次数已经比反面向上的次数多了 100 次以后，正面向上累积次数比反面向上的次数至少超出 100 万次的概率也是 1。

为了分析这些看起来互相矛盾的说法，我们更仔细地考察一下掷硬币的过程。我把一枚硬币掷了 20 次，得到如下结果：TTTTHTHHHHHHTTTHTTTH(H 代表正面向上，T 代表反面向上)，其中 11 次反面向上，9 次正面向上。根据大数定律，经过一段长的时间以后事件发生的频度应当非常接近于它们的概率。在这个例子中频度为 11/20＝0.55 和 9/20＝0.45——接近于 0.5 但并不等于 0.5。或许我掷出的结果看起来不够随机。你可能会对下面这样一类结果更满意一些：HTHHTTHTTHTHHTHTHHTT，因为这时正面向上和反面向上的频度均为 10/20＝0.50 了。这第二个结果除了使得数字刚好对头外，它看起来还显得更随机一些。然而实际上并非如此。

第一个结果看起来不大随机，因为它包含有一些长串的相同事件，例如 TTTT 和 HHHHHH 等，而第二个结果中则没有这一现象。然而，我们的直觉在这里起了误导的作用：随机序列常常呈现一些模



式和成团现象。不要因为看到这些现象而大惊小怪(除非掷硬币的结果是HHHHHHHHHHH……并一直像这样持续很长时间。在这种情况下,可以相当有把握地猜测这枚硬币的两面都是正面)。

假定你接连掷出了4枚硬币。掷出第1枚硬币后的结果不是正面向上就是反面向上(每种结果的概率都是1/2)。无论出现的是哪种情况,掷出第2枚硬币后的结果仍然为不是正面向上就是反面向上。如此类推。这样,对于抛掷4枚硬币来说,我们可以得到一棵“树”,其中有16条可能的路径。根据概率论,每条路径的概率均为$1/2\times1/2\times1/2\times1/2=1/16$。这一结果是有道理的,因为有16条路径,而且每条路径都是同等可能的。

注意,TTTT这条路径的概率为1/16,而其他路径(比如说HTHH)的概率也为1/16。这样,尽管HTHH看起来比TTTT更随机,但它们出现的概率是相同的。

另外,如果你把一枚硬币连掷4次,平均说来你会得到恰好两次正面向上。那么这是不是意味着得到两次正面向上和两次反面向上的可能性较大呢?并非如此。有16个不同的由H和T构成的序列,其中总共有6个序列含有两次正面向上的结果:HHTT,HTHT,HTTH,THHT,THTH,TTHH。因此,恰好有两次正面向上的概率为$6/16=0.375$。这一结果比未得到恰好两次正面向上的概率要小(后者的概率为0.625)。对于更长的序列,这一效果将变得更显著。

像这样一类研究使我们弄清了下面这个事实:事件的未来概率不受过去所发生的事情的任何影响,从这个意义上说,不存在任何平均律。然而,在一种相当令人感兴趣的意义上,可以说事件在长远的时期中仍倾向于趋于平衡。我们把正面向上超过反面向上的次数用图表示出来,也就是画出一幅表示每次掷硬币后正面向上与反面向上次数之差的曲线图。你可以把这幅图想象成是这样一条曲线:每当掷出一个正面向上时这条曲线就向上移动一步,而每当掷出一个

反面向上时这条曲线就向下移动一步。这样的图称为随机走动(random walk),其中连续的各步是随机地选择的。

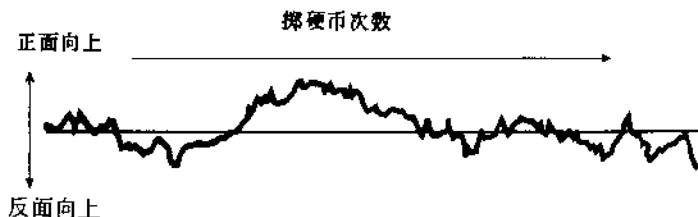


图 1　从随机走动曲线可以看出,正面向上的次数和反面向上的次数很少会扯平(在这种曲线中,"正面向上"对应于曲线向上移动一步,而"反面向上"对应于曲线向下移动一步)

图 1 是把硬币掷了 1 万次后所得的一个典型的随机走动图。像图中所示的那种极不平衡的行为是完全正常的。事实上,在掷 1 万次硬币的过程中,一侧在 9 930 次抛掷中领先而另一侧仅在 70 次抛掷中领先的概率大约为 1/10。

随机走动理论也告诉我们,正面向上次数与反面向上次数之差永远不返回到零(也就是说比如正面向上的次数永远居于领先地位)的概率为零。正是在这个意义上平均律成立——但是,如果你在赌掷硬币时是正面向上还是反面向上,那么它对于增进你的获胜机会起不了任何作用,因为你不知道这长长的老是正面领先的序列会有多长——你只知道它的确极有可能是非常之长。

假定你把一枚硬币掷了 100 次,其结果是 55 次正面向上,45 次反面向上——因此正面向上比反面向上的次数多 10 次。然后随机走动的理论就会告诉我们,如果你等待足够长的时间,这一差额将会被纠正(其概率为 1)。这是不是平均律呢?不。根据平均律通常的解释,情况并非如此。如果你事先选定了硬币抛掷的次数——比如说掷 100 万次——那么根据随机走动的理论,这 100 万次抛掷硬币的结果不受先前结果中那一差额的影响。此外,如果你通过再掷



100 万次硬币来做数目巨大的实验,那么你在合起来的总共 100 万零 100 次的抛掷中,平均说来将得到 50 万零 55 次正面向上和 50 万零 45 次反面向上这一结果。平均起来两者的差额依然存在。但是应当注意,正面向上的频度已经从 55/100 = 0.55 变为 500 055/1 000 100 = 0.500 005。平均律不是表现在把先前的差额去掉,而是表现在把它们"淹没"于巨大的数字中。

假设我不是掷一枚硬币而是掷一枚骰子,并统计每一面(从 1 至 6)出现的次数。假定每一面都有同等的机会出现,即概率均为 1/6。当我开始掷骰子时,每一面出现的累积次数是相等的——全都为零。通常,掷了几次骰子之后,这些数字便开始有所差别了。事实上,至少需要掷 6 次骰子之后各面出现的累积次数才有可能再次相等(即每面各出现一次)。那么,如果我把骰子一直不停地掷下去,到某个时候这 6 个面出现的次数再次扯平的概率是多大呢?我不知道确切的数值,因此这是"反馈信息"中需要填补的一个空白。不过我将向你证明它肯定不等于 1。

对于骰子问题,我们需要把随机走动的概念推广到多维的情形。例如,平面上最简单的随机走动发生在一个无限方格网的各个格点上。一个点从原点出发,每次向北、南、东或西走一步,朝这 4 个方向移动的概率是相同的,即各为 1/4。图 2 所示为一条典型的移动路径。三维随机走动则是在空间的立方体点阵上进行的,其情况与平面的随机走动相似,但它有 6 个走动方向,即北、南、东、西、上、下,每个方向的概率各为 1/6。

对于二维随机走动来说,也可以证明走动路径最终返回起点的概率为 1。Stanislaw M. Ulam——以前在洛斯阿拉莫斯国家实验室,其最有名的成就是与他人一起研制出氢弹——证明了在三维空间中,随机走动路径最终返回原点的概率为 0.35 左右(因此,如果你在沙漠中迷了路,并完全盲目地到处乱闯,你最终仍将返回绿洲。但

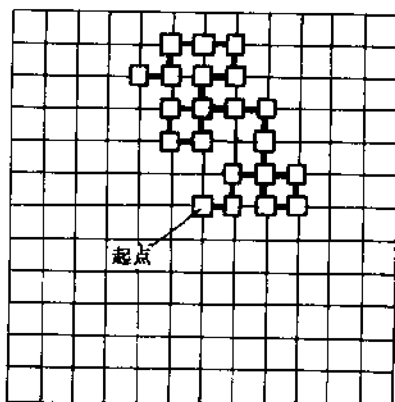


图 2　二维随机走动

是,如果你在空间迷了路,那么你凭运气返回地球的可能性只有1/3)。

假定我们用骰子各面上的数字给三维空间随机走动的 6 个方向编号——北=1,南=2,东=3,西=4,上=5,下=6。反复地掷骰子,并按规定的方向穿过空间点阵。在这个例子中,"返回原点"意味着 1 出现的次数和 2 一样多,3 出现的次数和 4 一样多,5 出现的次数和 6 一样多。因此,这种情况最终发生的概率为 0.35。那么所有 6 个数字出现的次数相等这一要求更高的事件的概率肯定小于 0.35。

甚至连最简单的一维随机走动也具有其他许多违背人的直觉的特征。假定你预先选定一个很大的抛掷硬币次数——比如说 100 万次,并观察是正面向上的次数领先还是反面向上的次数领先。那么平均说来,你觉得正面向上的次数应该在占多大比例的时间中领先呢? 你很自然地猜想应该是 1/2。实际上这一比例出现的可能性是最小的。最可能出现的比例是两个极端值:正面向上的次数要么从头到尾始终领先,要么任何时候都不领先!

3. 水泥艺术与复杂性理论

在 1997 年 12 月 11 日的一期《自然》杂志上，艺术史专家 Martin Kemp 介绍了伦敦一位名叫 Jonathan Callan 的艺术家创作的值得注意的风景作品。与一般的风景作品不同，Callan 的作品是雕塑而不是绘画，而且他的风景作品与地球看到的(事实上是任何已知的世界上所看到的)景观均不相似。这些风景作品是把水泥浇在一块多孔板上后所形成的三维形状。

Kemp 注意到 Callan 的雕塑作品和复杂性理论领域最新的研究工作之间存在某种关系。似乎有某些一般的原则支配着 Callan 的奇异的风景作品——例如，水泥的最高点位于距孔最远的地方。爱丁堡皇家天文台的一位天文学家指出，借助于一个更经典的数学分支——Voronoi 单元理论——可以了解 Callan 风景作品的奇妙的几何特性。他还解释了 Voronoi 单元是如何阐明近期的重大天文发现之一的(即物质在宇宙中的泡沫状分布)。如果存在一个数学、艺术和科学的统一的实例，那么这必定就属于此种例子了。

自从最初的洞窟绘画问世以来，艺术家们就一直依靠各种物理

和化学过程来创造他们的杰作。在古希腊和罗马,雕塑家们必须了解岩石是如何断裂的,而熔化的青铜又是如何流进铸模中的。文艺复兴时代的画家们则研究颜料的特性。传统艺术家所用的方法是控制这些物理过程,利用它们朝着预定的方向来创作出雕塑和绘画作品。Callan 是放弃了这种控制的一小批现代艺术家之一。他们的方法是让他们所用媒质的物理和化学过程来决定其作品的主要特性。

Callan 在开始创作每件雕塑作品时,第一步是在一块水平板上随机地钻一些孔。然后他把水泥粉末均匀地倒在板的表面上。有些水泥穿过孔掉了下去,有些水泥则在孔之间的板上堆积起来。这一雕塑品在吸收了空气中的水分后逐渐硬化。最终结果类似于月球表面的景观:锯齿状山峰围绕着陡峭的陨石坑。Callan 把这一雕塑同地球上的一个山脉作了比较:"一种既显得非常自然又显得高度人为的地形——全新的阿尔卑斯山。"

一种更适当的比拟可能是把 Callan 的作品比作许多沙堆。土木工程师们早就熟悉粒状材料(如沙、泥土和水泥粉末等)是如何堆积起来的。最简单、也是最重要的一个性质,是存在着所谓"临界角"(Critical angle)。粒状材料在堆积起来时可以保持一个最陡的斜坡而不坍落,此斜坡倾斜程度的大小与粒状材料的性质有关。这一斜坡与地面保持恒定的交角,这就是临界角。如果你把沙子堆积得越来越高——比如说让沙子成一条细沙流不断加到沙堆上——那么沙堆的坡度就越变越陡,直至达到临界角。达到临界角以后,再加上的沙将顺着沙堆滚下,引起一场小规模或大规模的崩落以恢复恒定的坡度。在这一最简单的模型中,这样所得的稳态形状是一个锥体,它的侧面的坡度恰好就是临界角。

复杂性理论家们研究斜坡达到这一形状的过程以及伴随着斜坡的生长而发生的崩落(不论大小)的性质。丹麦物理学家 Per Bak 发明了"自组织的临界状态"(Self-organized criticality)这个术语来描述



这类过程，并且他认为这些过程模拟了自然世界的许多重要特性(特别是进化。在进化中，崩落涉及的不是沙粒而是整个的物种，而相应的“堆”则是在一个由潜在有机物组成的假想的空间中)。

在 Callan 的作品中，每个孔周围的水泥粉末的结构正好同土木工程师们的圆锥形沙堆的结构相反。试考虑一块只有一个孔的水平板。水泥以这个孔为出发点向四面八方逐渐升起，其倾斜度就是临界角，这样就形成了一个圆锥形的凹陷，它的尖端指向下方，并且正好就在孔的中心上(见图 1)。这些倒置的圆锥面就是形成 Callan 的

图 1　参差不齐的环形山密布在 Jonathan Callan 的水泥雕塑作品之一的表面上(左图)。每个环形山的侧面的坡度与一个圆锥形水泥粉末堆的侧面的坡度相同(右图)



引人注目的风景作品的环形山。

但是，在有几个孔的情况下，又将出现什么样的几何形状呢？此时的关键问题在于，任何泻落的水泥粉末在穿过多孔板掉下时都将通过距离它的初始碰撞点最近的那个孔。因此，可以预测锥面环形山之间的边界将会出现在何处。把多孔板划分为若干个围绕着这些孔的区域，使得每个区域中的点到相应的孔的距离比到其他任何一个孔的都近。这一区域好比就是该孔的“势力范围”，只不过它不是球形而是多边形。假如多孔板是水平的，那么这些区域之间的边界正好就在相邻环形山的公共边界的下方。

可以用下面的方法来画出其中一个区域。选定任何一个孔，并画若干条线段把该孔的中心同其他所有的孔的中心连接起来（见图2的上图）。找出每条线段的中点，并从此点出发画另一条线与该线段相交成直角（也就是画出该线段的垂直平分线）。结果将得到由这些平分线构成的一个错综复杂的网络。找出以这一网络的组成线段为边界且包含所选定的那个孔的最小凸区域。该区域称为一个Voronoi 单元(Voronoicell)。每个孔都被一个唯一的 Voronoi 单元包绕着，所有的单元合起来就铺满了整个平面。Voronoi 单元还有其他不同的名称——如 Dirichlet 域和 Wigner-Seitz 单元等——因为这一数学概念曾在许多场合被重新发现。

因此，Callan 的环形山以相同的临界角成倒置锥面的形状耸起，且这些环形山彼此交会于他的那些孔所确定的各个 Voronoi 单元的边界上方。这种几何结构产生了一个令人愉快的结果——当两个坡面交会时，它们是在木板上方相同的高度上交会的，没有任何突然的间断。我们还可以推导出另外一个不那么明显的特征，即推导出一个环形山同与其相邻的环形山交会时所形成的山脊的形状。这种情况可以抽象地表示为两个以相同的倾斜角上升的倒置锥面。这两个锥面相交于它们的顶点之间连线的垂直平分线的上方，也就是相交



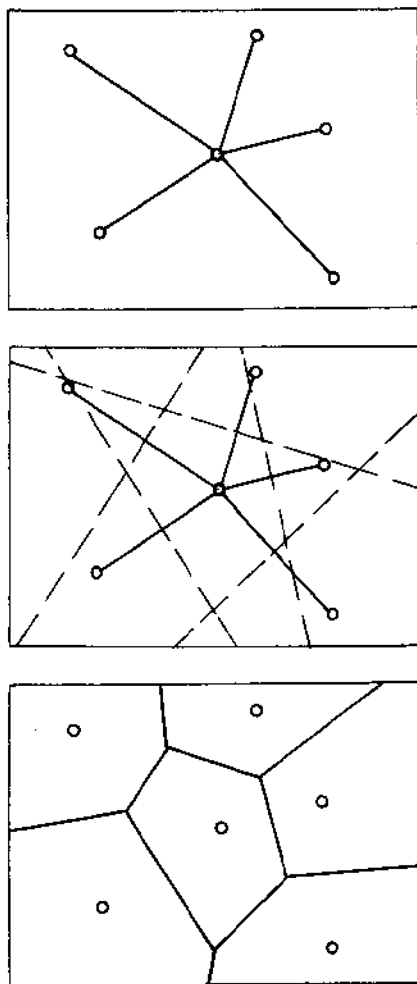


图 2 画出各个孔之间的垂直平分线，
就可以勾划出 Voronoi 单元

于 Voronoi 边界的上方。因此，它们的交线位于穿过该边界线的垂直平面内。一个垂直平面与锥面相交，会得到一条什么样的曲线呢?



古希腊人已经知道了这个问题的答案:双曲线。这一事实有助于说明 Callan 的风景作品为何具有那种相当参差不齐的外貌。

关于同天文学的关系又是怎么回事呢?这时我们想象三维空间中有一些点(相当于平面上的孔)。在平面上,两个点之间的连线的垂直平分线是一条直线,但在空间中,两点之间的连线被一个平面垂直平分。把某一给定点和其他所有点分别连接起来,并画出这些连线的垂直平分面。此时该给定点的 Voronoi 单元就是围绕这个点且被这些平面所界定的最小凸区域。现在这个 Voronoi 单元是一个多面体。天文学家们最近发现,宇宙中物质的大尺度分布类似于由这种多面体构成的一个网络。大多数星系团似乎都位于相邻的 Voronoi 单元的边界上。这种分布称为宇宙的 Voronoi 泡沫模型,因为它看起来有点像一个巨大的泡沫浴(bubblebath)。

这种分布同 Callan 的风景作品中水泥粉末的分布存在类似之处——这一比拟并非十全十美,但仍有启发作用。在 Callan 的雕塑中,水泥沿着 Voronoi 边界堆积得最高。而在三维空间中,类似的性质就是当宇宙膨胀时,物质将沿着同样的边界聚集。因此,这一简单的设想蕴含着引人注目的艺术、精巧的数学和关于宇宙结构的深刻的物理原理。

4. 骰子的奥秘

骰子是最早的赌博用具之一。本文中我将只讨论标准的现代骰子。这类骰子自然都是立方体,每一面上都有若干个点,其点数分别为1,2,3,4,5和6。相对两面上的点数之和均为7,这样骰子的6个面可以分为三对,即1与6,2与5,3与4。骰子的面恰好有两种配置方式具有这一性质,且这两种方式互为镜像。目前,西方制造的几乎所有骰子的点数为1,2,3的三个面沿着逆时针方向围绕着其公共顶点排列。有人告诉我说,在日本,具有这种手掷性的骰子用于除了麻将之外的所有游戏中。麻将这种游戏使用的是与其成镜像的骰子,从现在起,除非另有说明,我将使用西式骰子。

骰子常常是成对掷出,以便得到一个期望的总点数。首先假室骰子是"公平"的,这样掷出时每一面都有1/6的概率。为了计算某一总点数出现的概率,我们必须找出有多少种情形可以得到这一总点数。然后我们把这个数字除以36,即骰子对的总数(注意必须把两个骰子区别开来)。

想象一个骰子是红色而另一个骰子是蓝色有助于理解问题。这



样，比如说12这个总点数只能有一种情形，即红色骰子掷出6点，而蓝色骰子也掷出6点。因此总点数为12出现的概率为1/36。另外，总点数为11可以有两种情形得到，即红色骰子掷出6点，蓝色骰子掷出5点，或者红色骰子掷出5点，蓝色骰子掷出6点。这样总点数为11出现的概率为2/36，即1/18。

伟大的数学家和哲学家Gottfried Leibniz认为，掷出11点和12点的概率必定是相同的，因为在他看来只有一种情形掷出11这个总点数——也就是一个骰子掷出6点，而另一个骰子掷出5点。这一理论存在若干问题。最突出的问题或许是它同实验结果完全矛盾。实验结果表明，掷出11点的可能性为掷出12点的可能性的两倍。另外一个问题是，这一理论将导致一个不可靠的结论，即两个骰子掷出某一总点数——不管是多少——的概率小于1。

在有一种游戏——掷二骰赌博(craps)——中，对这些概率的直观感觉起着关键的作用。掷二骰赌博起源于19世纪40年代。在这种赌博中，一位参赌者(掷骰方)拿出一笔钱作赌注。其他参赌者则"跟进"(fade)，也就是赌他们自己选择的一笔数额的钱。如果跟进的钱的总额小于掷骰方开始时下的赌注，则他就把该赌注减少到与这一总额相等。然后掷骰方开始掷一对骰子。如果第一把掷出的骰子的总点数为7或11(称为"天然"点数(natural))，则他马上就赢了这场赌博。如果第一把掷出的骰子的总点数为2,3或12("craps")，则他就输掉了这场赌博。在其他情况下，掷骰方第一把掷出的总点数——即4,5,6,8,9或10——就是他们的"得分"。此时他必须继续掷下去，力争再次掷出一个得分，然后又掷出一个7("craps out")。如果能掷出这种结果，他就赢了所有赌注，否则他就输得精光。

根据前面提到的各个概率以及这一赌博的规则，可以计算出掷骰方获胜的机会为244/495，即49.3%左右。这比胜负机会均等的概率(50%)刚好小一点。职业赌棍可以通过两种方法把这一微小的



不利条件转化为优势。一种方法是接受或拒绝与其他参赌者的各种"附带赌"(即超过一般赌注的打赌)。另一种方法则是弄虚作假,在赌博中用掩人耳目的巧妙手法使用做了手脚的骰子。

可以有多种方法在骰子上做手脚。骰子的各面可以巧妙地加以修削,使它们的各个角不成直角,也可以用重物给骰子"灌铅"。这两种方法都可以使骰子掷出某些点数的可能性大于另一些点数。更富有戏剧性的做假手法是用"顶骰"(top)和"底骰"(bottom)来代替标准的骰子。这两个骰子的各面只有 3 个不同的点数(相对各面的点数相同)。由于任何一位参赌者在任一时候最多只能看到一个骰子的 3 面,而且所有相邻的面的点数均不相同,所以粗看一下似乎没有什么不正常的情况发生。然而,不可能保证所有顶点上各个面都按标准次序排列。事实上,如果在某一顶点上点数为 1,3,5 的 3 个面按反时针方向排列,则在相邻顶点上这 3 个面就必定按顺时针方向排列。

在掷双骰赌博中,顶骰和底骰可用来达到各种不同的目的。例如,使用一对 1—3—5 的骰子,永远也不可能掷出 7 这个总点数,因此用这类骰子一位参赌者永远也不可能赢(crap out)。把一个 1—3—5 的骰子和一个 2—4—6 的骰子合起来用,则不能得出偶数的总点数,因此用这样两个骰子一位参赌者不可能掷出 4,6,8 或 10 这些总点数。如果要使这些作弊行径不被人察觉,则顶骰的使用不可太多——假如老是掷出偶数的总点数,那么甚至连最无经验的参赌者也会起疑心的。

许多戏法或聚会上玩的把戏都使用骰子。其中相当多的戏法利用了骰子的相对各面的点数之和为 7 这一条规则。Martin Gardner 在他的著作《数学魔术》中介绍了一个戏法。魔术师转过身去,请一位观众掷 3 颗标准骰子,然后把朝上的各个面的点数加起来。接着魔术师请这位受骗者拿起任何一个骰子,把其朝下的一面的点数加

在前面得到的总数上。最后,这位观众把这个骰子再掷一次,把朝上的一面的点数加在第二个总数上(他必须自己记住所有这些总数)。现在魔术师转回身来,随口报出结果是多少,尽管她并不知道该观众选择的是哪一个骰子。

奥妙何在呢?假定这些骰子朝上一面的点数分别为a,b和c,且该观念选择的是a骰。最初的总和是$a+b+c$,在这一总和中加上$7-a$,就得到$b+c+7$。然后把a骰再掷一次,得到的点数为d,于是最终结果为$d+b+c+7$。接着魔术师看看这三个骰子,它们朝上一面点数的总和为$d+b+c$,这样魔术师只须很快地把这3个数加起来再加上7就大功告成了。

英国难题专家Henry Ernest Dudeney在他的著作《趣味数学》中介绍了一种不同的把戏。魔术师仍然转过身去,请一位观众掷了个骰子。但现在她是让这位受骗者把第一个骰子的点数乘以2再加上5,把这个结果乘以5后再加上第二个骰子掷出的点数,接着再把此结果乘以10,最后再加上第三个骰子掷出的点数。在得知这一最终结果后,魔术师就立刻报出这三个骰了掷出的点数各为多少。自然该观众得出的最终结果是$10(5(2a+5)+b)+c$,即$100a+10b+c+250$。因此魔术师只须从这个结果中减去250,剩下的三位数中的三个数字就分别是三个骰子所掷出的点数了。

其他骰子问题则涉及一些改动了的骰子,它们具有非标准的点数。例如,读者是否能想出一种方法,只用0,1,2,3,4,5或6这几个数字来给一对骰子规定点数,使得这对骰子掷出后其总点数之和的所有各种可能情形(从1到12)出现的机会一样大(答案见本文末尾)?或许最不合符人类直觉的骰子现象是所谓“非可递骰子”。做3个骰子A、B、C,其各面上的点数如下:

A:334488　B:115599　C:226677

在掷了许多次以后,骰子B掷出的点数平均说来将胜过骰子A



掷出的点数。事实上,骰子B掷出的点数比骰子A掷出的点数大的概率为5/9。类似地,骰子C掷出的点数比骰子B掷出的点数大的概率也为5/9。那么骰子C掷出的点数平均说来显然该比A掷出的点数大了,对吧?不,恰恰相反,骰子A掷出的点数比骰子C掷出的点数大的概率为5/9。附图阐述了上述说法的理由。你可以用这样一套骰子大赚其钱!让你的赌博对手任挑一个骰子,然后你再选一个可以压倒它的骰子(掷了许多次以后,你的骰子点数超过对手骰子点数的概率大于1/2)再重复这样赌下去。你将在所有赌局的55.55%中获胜。但你的对手却可以自由选择他认为"最佳的"骰子!

A	3	4	8
1	A	A	A
5	B	B	A
9	B	B	B

B	1	5	9
C			
2	C	B	B
6	C	C	B
7	C	C	B

C	2	6	7
A			
3	A	C	C
4	A	C	C
8	A	A	A

图　本图所示的非可递骰子(分别称为A、B和C)具有一种奇妙的性质:平均说来,骰子B掷出的点数大于骰子A,骰子C掷出的点数大于骰子B,但骰子A掷出的点数又大于骰子C。A的各面上只有3,4或8这三种点数,B的各面上只有1,5或9这三种点数,而C的各面只有2,6或7这几种点数。本图显示了这些奇怪的骰子中的任何一个与其他某个骰子进行较量时的可能结果

不过要当心:不要过于信赖概率论而忘了把赌博的规则搞得非常严密。Ivar Ekeland在他那本绝妙的小书《裂开的骰子》(芝加哥大学出版社,1993)中讲了这样一个故事。两位北欧国王通过掷骰子来决定一个有争议的小岛的归属。瑞典国王掷出两个骰子,其点数都是6。于是他得意地说,不会有比这个点数更好的结果了,所以挪威的奥拉夫国王索性干脆认输为妙。奥拉夫国王则咕哝了几句,不外是说他也可能掷出两个6点,接着他掷出两个骰子。结果,一个骰子



的点数为6，另一个却裂成两块，一块的点数为1，另一块的点数为6。总点数为13！这整个故事的目的是要证明你认为可能的事情取决于你如何建立问题的模型。

当心，有几个好挖苦的人认为奥拉夫国王炮制了这整个骗局。

答案：要使一对骰子从1到12所有情形下的所有总点数之和相等，则一粒骰子的点数应为1，2，3，4，5和6；另一粒骰子的点数，应为0，0，0，6，6，和6。

5. 高尔夫球的晶体学

谁能想到看来不起眼的高尔夫球中竟会蕴含许多数学奥秘呢?高尔夫球的生产厂家采用多种多样的凹点分布图案,几乎所有这些图案都是高度对称的。而这正是数学大有用武之地的地方。晶体学家们运用这同一类数学知识来分析晶体点阵的对称性。

对称性和凹点的多少是有关系的,因此高尔夫球有幻数个凹点——比如252,286,332,336,360,384,392,410,416,420,422,432,440,480,492和500个凹点——的情形多于其他数目凹点的情形。所有这些凹点数都能在市售高尔夫球上找到。有一厂家曾生产过有1 212个凹点的高尔夫球,不过那不是真正用来打的,只是为了显示该厂家制造高尔夫球的技术如何高明而已。

凹点必须以相当均匀的方式分布在整个高尔夫球的表面上。否则高尔夫球在飞行时往往会突然转向,因为空气阻力与凹点的密度有关。这样,设计高尔夫球的问题就同把微小的圆盘均匀地分布在一个球的表面上这一数学问题密切联系起来了。但是,也有一些实际的因素必须考虑。特别是,高尔夫球是由两个半球模压成的,因此



任何情况下都必定至少有一条“分界线”，即球面上不经过任何一个凹点的大圆。为了确保空气动力平衡，高尔夫球的大多数设计方案都采用一对称布置的多条伪分界线。这些线条是了解高尔夫球的对称性的极好线索。

在通常的语言中，“对称性”一词的用法是相当含混的，一般指某种东西排列得很匀称或巧妙。在数学上，对称性的定义要严密得多。它不涉及形状，而是涉及变换，即移动某个图形的数学规则。如果物体在经过变换之后与其变换之前的形状完全重合，那么此变换就是该物体的一种对称性。例如，正方形有 8 种对称性(不考虑弯曲或拉伸之类的变形)，这 8 种对称性包括(逆时针)旋转 90 度，180 度和 270 度，以及关于正方形的轴线或对角线的镜面反射。第 8 种对称性是所谓“平凡”变换，它使正方形映射到自身。

一个物体的所有对称性构成一个“群”：如果连续施行两种对称变换，其结果也是一个对称(称为这两个对称变换的复合或积)。例如，如果把正方形旋转 90 度后再旋转 180 度，那么结果就是旋转了 270 度，这也是该群的一个元素。这一“群性质”是我们之所以要引入平凡对称性的理由之一。如果把平凡对称性略去不计，则两种对称性之积就可能不属对称性之列了。例如，旋转 90 度与旋转 270 度之积为旋转 360 度。由于旋转 360 度的结果是正方形的每一点又回到其原来位置，因此这不过是平凡对称性的另一种说法而已。

群的结构使我们可以把对称物体分为不同的类型。如果两个物体的对称变换群有相同的抽象结构，则它的对称类型就是相同的。比如说，一个正方形和另一个四个角被弄圆了的正方形就属于同一种对称类型。

在二维和三维空间中，最重要的对称类型是旋转和反射。二维空间中的旋转就是使物体绕着一个固定点转动某一角度。在三维空间中则是使物体绕着一根固定轴转动某一角度。二维空间中的反射



是交换在一根固定的“镜面反射轴”两侧处于相似位置上的两个点，三维空间中的反射也是进行这种交换，但用固定的反射面代替了反射轴。

对称群的一个重要特征是它的阶，即它包含的不同对称变换的数目。这样，正方形的对称群的阶就是 8。三维空间中的对称性最高的物体中包括人们熟知的 5 种正多面体，即正四面体、正六面体（立方体）、正八面体、正十二面体和正二十面体。它们的对称群的阶分别为：

- 正四面体（有 4 个三角形的面）：24
- 立方体（有 6 个正方形的面）：48
- 正八面体（有 8 个三角形的面）：48
- 正十二面体（有 12 个五边形的面）：120
- 正二十面体（有 20 个三角形的面）：120

在每种情况下，对称群的阶数都等于其面数的 2 倍再乘以每一面的边数。例如，正十二面体的阶数为 $2\times12\times5=120$。这一关系是这些立体的规则几何形状所造成的结果。为了看出其原因，可以选择某一个面：转动正十二面体，就可以选择它的 12 个面中的任何一面。在使这一面保持不动的同时，可以使正十二面体旋转到 5 个位置中的任何一个上。这样就得到 $12\times5=60$ 种旋转对称性。但是选定的这个面也可以在进行旋转前被反射，因此使得对称性的数目又翻了一番，最终达到 120。

注意立方体和正八面体有相同的对称阶数，而正十二面体和正二十面体的对称阶数也相同。这是对偶性的结果。正八面体各面的中点构成一个立方体的顶点，而立方体各面的中点也构成一个正八面体的顶点。这就意味着正八面体和立方体有相同的对称群。正十二面体和正二十面体的情况也是一样的。

三维空间中对称群的最高阶数并不是 120。然而，超过 120 阶



的三维对称群必定有无穷多种对称性。但是,由于凹点的数目有限,因此一个有凹点的高尔夫球的对称群其阶数必定为有限,这样它的对称性阶数最多只能为120。

当然,高尔夫球的凹点数目可以大于120。早期高尔夫球采用两种不同的设计图案。在第一种图案中,凹点沿着与赤道分界线平行的纬度线排列,通常有392个凹点,这种高尔夫球具有五重旋转对称性,其对称轴为穿过高尔夫球的南极和北极的轴线(见图),但由于某些莫名其妙的原因,常常去掉了8个凹点(b)。第二种图案的对称类型与正八面体相同。正八面体是由8个等边三角形构成的(例如把两个底面为正方形的棱锥底对底拼起来就可以得到正八面体)。典型的正八面体高尔夫球(c)有336个凹点和3条分界线,这些分界线彼此相交成直角,就像地球的赤道与零度及90度上的两根经线一样。

一种更深奥的设计方案具有二十面体的对称性及6条分界线。一个二十面体由20个等边三角形构成,每个顶点上有5个三角形。在建筑物的网格球顶(如雷达设施的顶部)以及许多病毒的蛋白单元中也可发现此类结构。例如,腺病毒就有252个蛋白单元,排列成一个“二十-十二面体”,其各面为正六边形或等边三角形,每个顶点上有5个面。Uniroyal公司生产的一种高尔夫球正好就是这种结构(e):说来也怪,它的252个凹点是五棱锥的形状。

1973年,Titleist公司推出一种高尔夫球(f)。乍看起来它的形状为二十面体。但它的对称性是不完整的。制作的第一步是作成正二十面体,使凹点填满每个三角形面。但是这一特殊的排列方式没有分界线,因此把沿赤道的一行凹点折起来,使其间出现一道清晰的间隙。这种图案破坏了二十面对称性。但却得出了一种完全实用的高尔夫球。



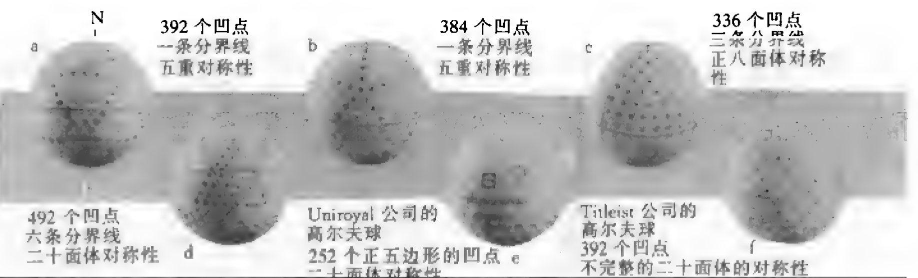


图　高尔夫球表现了有限对称群的复杂性

6. 法庭上的概率论

数学正在侵入法院的审判室。陪审团通常会得到这样的嘱咐：如果他们“有合理的把握”相信犯罪嫌疑人犯下了某一罪行，就可以判他有罪。这种嘱咐是定性的，它与陪审员心目中什么才算“合理”有关。未来的文明社会或许会尝试使“有罪”定量化，也就是用一台法院计算机来代替陪审团，对证据进行权衡，并计算出有罪的概率。但是如今我们还没有法院计算机，这样陪审团就不得不设法掌握概率理论了。

原因之一在于DNA证据的使用日益广泛。DNA特征分析学还是一门比较新的学科，因此对DNA证据的解释有赖于概率的评估。在传统的指纹分析术最初被采用时，可能也出现过类似的问题，不过那时律师们或许还没有现在这样成熟；无论如何，人们不再以概率上的理由对指纹证据进行辩驳。

Robert A. J. Matthews指出，法庭案例中的一项远为传统得多的证据来源（也就是犯罪嫌疑人的交代）应当用概率论加以分析。西班牙第一任宗教法庭庭长Tomas de Torguemada认为，犯人的交代就



是其有罪的充分证据，即使这交代是逼供信的结果（通常也确是如此）。Matthews 的最令人感到意外的一项结论是，在某些情况下，犯罪嫌疑人的交代更有利于认为他有罪的看法而不利于认为他有罪的看法（Matthews 称这个结论为“审讯者的谬误”）。

Matthews 的理论为怀疑受审讯的恐怖分子的交代提供了理由，除非有其他证据为其交代作佐证（恐怖分子在审讯中是很顽固的）。当代的司法实践对于通过逼供得出的交代是非常怀疑的。在英国，一系列宣布恐怖分子有罪的著名判决都以口供证据为依据，但由于怀疑口供的真实性，这些判决已被推翻。

解释 Matthews 的结论所需要的主要数学概念是条件概率的概念。假定史密斯先生和史密斯太太告诉你说他们有两个孩子，其中一个是女孩。那么另一个孩子也是女孩的概率是多大?

也许你会不假思索就回答说另一个孩子不是男孩就是女孩，其概率各为 1/2。然而这个问题中的性别分布存在着四种可能的情形：BB，BG，GB 和 GG，其中 B 和 G 分别代表男孩和女孩，而字母的次序则表示出生的先后顺序。每种组合方式都有同等的可能性出现，因此其概率各为 1/4。其中恰好有三种情形（BG，GB 和 GG）是这个家庭至少有一个女孩，而这三种情形中仅有一种情形（GG）是另外一个孩子也是女孩。因此，在已知至少有一个女孩的情况下，两个孩子均为女孩的概率实际上是 1/3。

另外，假定史密斯夫妇告诉你说他们的最大的孩子是女孩。那么小的一个孩子也是女孩的概率为多大呢？这时，可能的性别分布只有两种情形，即 GB 和 GG，而只有 GG 才是最小的一个孩子也是女孩的情形。因此这个概率就成了 1/2。

这一类概率被称为条件概率，即某一事件在另外一个事件确实已经发生的条件下出现的概率。正如史密斯夫妇的孩子们所表明的那样，使用条件概率时需要规定一个背景，它可能对计算出的概率有



强烈的影响。

为了看出这类问题的复杂性，假定有一天你看见史密斯一家在他们的花园里。一个孩子很明显是女孩，另一个孩子被他们家的狗遮住了一部分，因此不能肯定是男孩还是女孩。那么史密斯夫妇有两个女孩的概率是多大?

你可以认为这个问题同上面所述的第一种情形完全一模一样，因此其概率应当是 1/3。但你也可以认为现在提供给你的信息是“不在逗狗玩的那个孩子是女孩”。这一陈述同上面的第二种情形一样，把一个孩子同另一个孩子区分开来，因此其答案是 1/2。而史密斯夫妇知道逗狗玩的那个孩子是威廉，因此他们会说他们有两个女孩的概率为零。究竟哪一种说法正确呢?

这个问题的答案取决于对背景的选择。你是否是从有许多个不同的家庭，而每个家庭的每个小孩都同狗玩的这样一类环境中随机取的样? 或者你是否从其中只有一个小孩经常同狗玩的家庭中随机取的样? 或者你是否只是在考察一个特定的家庭(在这种情况下概率是完全错误的模型)?

为了解释统计数据，需要了解概率的数学及运用概率的背景。很久以来，律师们一直在肆无忌惮地钻陪审员们数学知识不多的空子。DNA 特征分析——现在已被法庭所充分了解——中的一个例子是“检察官的谬误”。DNA 特征分析是莱塞斯特大学的 Alec J. Jeffreys 在 1985 年发明的，它依据的是人体基因组中所谓的“可变数目连续重复”区(Variable number of tandem repeat，缩写为 VNTR)。在每一个这样的区中，某一特定的 DHA 序列重复许多次。人们普遍认为 VNTR 序列可以作为个体的独特的标识。

为了用于法庭上，科学家们使用分子生物学的标准方法来寻找两份 DNA 样品——一份与犯罪现场有关，另一份取自犯罪嫌疑人——的若干不同 VNTR 区之间的符合情况。如果找到了足够多的



符合之处，那就是表明两份样品来自同一个人的令人信服的统计学证据。

检察官的谬误指的是把两种不同的概率混淆起来。“符合概率”回答的是这样一个问题：假定某人是无罪的，则他或她的 DNA 与犯罪现场取样相符合的概率有多大？但是与法庭有关的问题则是：“假如两份 DNA 样品符合，则犯罪嫌疑人无罪的概率有多大？”这两个问题的答案可能大大相同。

答案不同的起因也是在于背景。在第一种情况下，我们想象把个人置于为了科学上的方便起见而选择的某一较大群体中。在第二种情况下，他或她则是被置于一个界定不那么严格但却更有关系的群体中——也就是由那些从道理上说有可能犯下此项罪行的人组成的一个群体。

在这样的情况下条件概率的运用服从于一个据信是英国人 Thomas Bayes 发明的定理(贝叶斯定理)。设 A 与 C 为事件，其发生的概率分别记为 p(A)和 p(C)。在 C 确实已发生的条件下，A 发生的概率记为 p(A|C)，用“A&C”表示“A 与 C 都已发生”。于是贝叶斯定理告诉我们 p(A|C) = p(A&C)/p(C)。

例如，在史密斯的孩子这个例子中(第一种情形)，我们有：

C = 至少有一个孩子是女孩

A = 另外一个孩子也是女孩。

p(C) = 3/4

p(A&C) = 1/4

这是因为，A&C 也就是“两个孩子都是女孩”这一事件，即 GG。然后，根据贝叶斯定理，在已知两个孩子中有一个是女孩的情况下，可以算出另一个孩子也是女孩的概率为(1/4)/(3/4) = 1/3，这是我们先前已经得出的结果。类似地，对于第二种情形，贝叶斯定理给出的答案为 1/2，这也是先前已得出的结果。



在用于口供证据上时,Matthews 规定了下面的记号:

A=犯罪嫌疑人有罪

C=他或她已经交代罪行

按照与贝叶斯定理有关的推导过程的通常作法,Matthews 把 p(A)取为犯罪嫌疑人有罪的"先验概率",也就是根据在犯罪嫌疑人交代罪行之前已获得的证据所确定的有罪概率。设 A′表示事件 A 的反面,既"犯罪嫌疑人无罪"。

然后(根据框图内所述之计算过程),Matthews 得出了公式 $p(A|C)=p/[p+r(1-p)]$。为了使公式简洁,我们用 p 表示 p(A),而用 r 表示 $p(C|A')/p(C|A)$,并称 r 为交代比率。这里 $p(C|A')$是一个无罪的人招供有罪的概率,而 $p(C|A)$则是一个犯罪的人招供有罪的概率。因此,如果一个无罪的人招供有罪的概率比一个犯罪的人招供有罪的概率小,则交代比率小于 1。

如果要使犯罪嫌疑人的交代增大其有罪的概率,则我们就希望 $p(A|C)$大于 p(A),而 p(A)等于 p,这样,我们需要 $p/[p+r(1-p)]>p$,稍加化简后得:$r<1$。这就是说,仅当一个无罪的人招供有罪的可能性比一个犯罪的人招供有罪的可能性小时,犯罪嫌疑人招认罪行才会增大其有罪的概率。

上述结果的含义是,在某些时候,犯罪嫌疑人招认有罪可能会减小其有罪的概率。事实上,只要无罪的人招认有罪的可能性比犯罪分子招认有罪的可能性大,这种情况就会发生。在涉及恐怖分子的案件中,可能会出现这类情况。心理分析指出,那些更易受他人暗示影响的人,或更易依从他人的人,在接受审讯时招认有罪的可能性较大。但这些说法一般不适用于顽固不化的恐怖分子,因为他们都受过抗拒审讯方法的专门训练。当英国法院作出现在已被推翻的那些有罪判决时,很有可能正是发生了这种情形。

贝叶斯分析还揭示出证据的其他某些与直觉矛盾的特征。例



如,假定最初的有罪证据(X)随后又得到了有罪的补充证据(Y)。陪审团几乎总是会认为现在有罪的可能性增大了。但是有罪的概率并不是以这种方式递增起来的。事实上,仅当在已知旧的证据且犯罪嫌疑人有罪的情况下出现新证据的概率大于在已知旧的证据且犯罪嫌疑人无罪的情况下出现新证据的概率时,新证据才会增大有罪的概率。

当起诉案件取决于犯罪嫌疑人的交代时,可能会出现两种完全不同的情况。在第一种情况中,设 X 为犯罪嫌疑人的交代,而 Y 为根据这一交代所发现的证据,比如说在犯罪嫌疑人所交代的地方找到了受害人的尸体。由于无罪的人不大可能提供这类线索,因此贝叶斯分析显示有罪的可能性增大了。这样,需要犯罪嫌疑人作出真实交代才能获得的补充证据就增加了其有罪的概率。

另外,X 可能是发现了尸体,而 Y 则是犯罪嫌疑人随后所作的交代。在这种情况下,尸体提供的证据与犯罪嫌疑人的交代无关,因此不能证实交代。但是,并不存在与"审讯者的谬误"相类似的"尸体发现者的谬误",因为我们很难认为,仅仅由于知道了一具尸体已经被找到,无罪的人招认有罪的可能性就比有罪者招认有罪的可能性更大。

当然,主张所有将来可能当陪审员的人都参加通过贝叶斯推理的培训是愚蠢的,但是,法官可以就某些简单的原理对他们进行指导,这看来却是一个完全可行的建议。此外,同样的设想也适用于 DNA 特性分析,但仅是在对陪审员来说要直观得多的场合中。迅速地回顾一下审讯者的谬误可能是阻止律师们对 DNA 证据提出荒谬诊断的绝好方法。

Matthews 公式的推导

根据贝叶斯定理我们有:$p(A|C) = p(A\&C)/p(C)$

以及类似地:

$p(C|A)=p(C\&A)/p(A)$

但是 C&A = A&C，所以我们可以把这两个公式结合起来而得到：

$p(A|C)=p(C|A)p(A)/p(C)$

此外，由于 A 与 A′这两个事件中必定有一个会发生，但不可能同时发生，因此有：$p(C)=p(C|A)p(A)+p(C|A')p(A')$

最后，我们有 $p(A')=1-p(A)$

把上述结果综合起来，可得：

$p(A|C)=p(A)/[p(A)+P(C|A')p(A')/p(C|A)]$

如果我们用 p 代替 p(A)，用 r 代替 $p(C|A')/p(C|A)$ 就得到 $p(A|C)=p/[p+r(1-p)]$

7. Juniper Green 游戏

玩这个游戏需要准备 100 张卡片,用 1 到 100 的数字分别给这些卡片编号。把卡片按编号顺序正面朝上放在桌子上,比如可以放 10 行,每行 10 张卡片,这样游戏者就很容易找到所要找的卡片。这个游戏的规划如下:

1. 两个游戏者轮流从桌子上取一张卡片。取出的卡片不再放回,也不能再次使用。

2. 除了第一步以外,以后所取的每张卡片上的数字都必须是另一位游戏者上次所选择数字的整数因数或倍数。

3. 无法再选择一张卡片的一方为输家。

还有最后一条规则使这个游戏值得一玩。读者应当记得,素数除了它本身和 1 以外没有其他任何因数。凑巧的是,如果一位游戏者选择了一个大于 50 的素数,那么另一位游戏者就非输不可。假定 Alice 同 Bob 玩这种游戏,Alice 先走。她选择了 97,于是 Bob 只能选 1。接着 Alice 又选另一个大于 50 的素数,比如说 89。这时 Bob 已经用了卡片 1,因而走投无路了。为了避免这种令人扫兴的策略,该



游戏还有下面这样一条规则：

4. 游戏的第一步必须选择一个偶数。

即使游戏以偶数开局，大素数仍然影响着双方的选择。特别是，如果一位游戏者选择了卡片1，那么只要对手没有睡大觉，他就肯定输了。比如说Bob选了1，而Alice则选择一个大素数——97(注意97是必定可用的，因为这个数只有在另一位游戏者选了1时才能被选上)。这样Bob就束手无策了。因此，这一游戏实际上以一位游戏者不得不选择卡片1而告终。

下面示出了一局游戏的经过，此时游戏双方没有过多考虑采用巧妙的策略。我建议读者到此暂且不要再读下去，而是动手准备一套卡片并玩一下这个游戏。我将分析一下只有40张卡片时(分别用1到40的数字编号)这同一种游戏的玩法。这分析也将使你获得如何玩100张卡片游戏的某些一般线索。幼儿可以用编号为1到20的卡片来玩。为简短起见，我将把有n张卡片的Juniper Green游戏称为"JG-n"，并将找出JG-40的获胜策略。

当然，有些开局的走法很快就会输掉。试看下面这个例子：

步数	Alice	Bob
1	38	
2		19
3	1	
4		37
5	输掉	

第一步选34，也会得到同样的结果。其他某些数最好也应尽量躲开。例如，假定Alice昏头昏脑地竟然选了35，Bob则乘机选25作为报复。这样Alice别无他途，只能选1。但这就注定她输掉了这局游戏(注意25必定是仍然可以使用的；因为只有当另一位游戏者上一步选的是1或5时这个数才能被选用)。

Alice显然应采取的策略是迫使Bob不得不选5。她能做到这一



点吗？好了，如果 Bob 选 7，那么她可以选 35，这样 Bob 就只能选 1 或 5，而这两者都将使 Bob 输掉。此计不错，但她能够迫使 Bob 选 7 吗？答案是肯定的：如果 Bob 选 3，则 Alice 可以选 21，这就迫使 Bob 选 7。很好，但她如何使 Bob 选 3 呢？嗯，如果 Bob 选 13，那么 Alice 选 39。Alice 可以像这样一直推算下去，得出一连串假想的选择过程，在每一步都迫使 Bob 不得不选某个数，最终不可避免地掉进失败的深渊。

步数	ALICE	BOB	解说
1	48		根据第 4 条规则的要求选一个偶数
2		96	此数为 Alice 选择的数的 2 倍
3	32		此数为 Bob 选择的数的 1/3
4		64	Bob 不得不选择 2 的乘方
5	16		Alice 也作此选择
6		80	
7	10		
8		70	
9	35		此数为 Bob 选择的数的一半
10		5	只能选 7 或 5(再有就是选 1，从而输掉游戏)
11	25		
12		75	只能选 50 或 75
13	3		
14		81	
15	9		只能选 27 或 9
16		27	这一步不妙
17	54		只能选 54(否则就是选 1，从而输掉游戏)
18		2	只能选 2
19	62		受大素数策略启发而得出的走法
20		31	只能选 31
21	93		只能选 93，但这一步选得好
22		1	不得不选 1，从而输掉了，因为……
23	97		大素数策略



然而 Alice 能够从一开始就引诱 Bob 钻进这样一个圈套吗？游戏开始时的几步必定与偶数有关，因此卡片 2 有可能起着一种关键的作用。事实上，如果 Bob 选 2，则 Alice 可以选 26，从而迫使 Bob 落入不得不选 13 这样一个陷阱。于是我们终于抓到了问题的关键：Alice 如何才能迫使 Bob 选 2？

如果 Alice 一开始就选 22，则 Bob 可以有两种选择。如果他选 2，则他就掉进了上面所说的每一步都身不由己任人摆布的圈套中。但他也可以选 11。如果他选 11，则 Alice 也可以有两种选择：选 1（从而输掉游戏）或选 33。如果她选 33，则因为 11 已经用过，所以 Bob 只能选 3，于是 Alice 就可以赢得游戏了。下表列出的走法概括了 Alice 所用的策略，两组走法分别表示 Bob 可以选择的方案（假定两位游戏者自始至终都避开了 1）。

步数	ALICE	BOB	ALICE	BOB
1	22			
2		11		2
3	33		26	
4		3		13
5	21		39	
6		7		3
7	35		21	
8		5		7
9	25		35	
10		LOSES		5
11			25	
12				LOSES

至少还有另一种开局选择——选 26——使 Alice 能最终获胜。此时游戏按同一类方式进行下去。但有几步互相换了位置，如下表所示：



步数	ALICE	BOB	ALICE	BOB
1	26			
2		13		2
3	39		22	
4		3		11
5	21		33	
6		7		3
7	35		21	
8		5		7
9	25		35	
10		LOSES		5
11			25	
12				LOSES

这里起关键作用的是素数 11 与 13。如果第一步 Alice 选择的数是一个素数的 2 倍(22 或 26),则 Bob 只能选择 2——此时 Alice 就肯定能获胜了——或选择该素数。但 Alice 接下来选择该素数的 3 倍,迫使 Bob 只能选择 3,这样 Alice 依然将获胜。Alice 之所以能赢,是因为 11 或 13 这两个素数小于 40 的倍数除了 2 倍的倍数外,其他就只有一个倍数了(即 33 或 39)。这些"中间素数"的值在卡片数目的 1/4～1/3 之间,它们保证了 Alice 将赢得游戏。

除了 22 或 26 外,是否还有其他开局选择也可保证游戏者得胜呢? 这个问题要留给读者去探讨了。此外,读者现在已有了充分的准备,可以分析 JG-100——甚至可以分析令人望而生畏的 JG-1 000。是否存在一种策略使先走者必定获胜呢?

最后,我们可以把这个问题推广为它的最一般的形式。试考虑 n 为任意整数时的 JG-n 游戏。由于此游戏不允许平局,因此由对策论可知,Alice——她是先走者——或 Bob 可以有获胜策略,但不可能两人都有。设 n 在 Alice 有 JG-n 游戏的获胜策略的情况下为"主要的",而在 Bob 有 JG-n 游戏的获胜策略的情况下为"次要的"。读者能描述哪些 n 是主要的,哪些 n 是次要的吗?



当n很小时，稍加计算就可以知道1，3，8和9是主要的，而2，4、5，6和7是次要的。当n为100时情况又如何呢？n为任意值时又如何？有人能找出任何规律来吗？或者是否能解决这全部问题？

8. 字母幻方

幻方就是由若干数字构成的一个方阵，其各行、各列及主副对角线上的数字之和均相等。幻方很久以来一直是趣味数学的一个重要内容。根据中国传说，最简单的幻方

4	9	2
3	5	7
8	1	6

是公元前23世纪时上天通过一块龟板泄露给禹帝的，这个幻方的公共和（即"幻方常数"）为15，而其大小（即"阶"）为3。所有阶数大于3的幻方都是存在的，此外还有不计其数的推广形式，如幻立方，幻六边形，幻八边形，幻圆等等。

有人可能以为，关于这类数学结构，能说的早就已经说完了，然而，10年前，Lee Sallows发明了一种全新的幻方，即字母幻方。Sallows是一位搞文字游戏的老手，他专门从事把这类游戏同趣味数学结合起来的工作。

下面是一个例子：

five	twenty-two	eighteen
twenty-eight	fifteen	two
twelve	eight	twenty-five

如是把这些单词转换为数字,就成了一个通常的幻方(其幻常数为45)。但是,如果我们数一下每个单词中有多少字母(连字符不算),则得到:

4	9	8
11	7	3
6	5	10

这也是一个幻方,其幻常数为21。

Sallows研究出这类数学结构的一个一般性的理论。作为第一步,他定义log(x)—x这个数的"logorithm"—为x的书面表达式中所含的字母的个数,(在这里,Sallows把希腊语中的logos—意为单词—与表示"数"的arithnos巧妙地结合起来,使logarithm有了双关的含义)。同一个数字在不同的语言中其logorithm也不相同;眼下我们暂且只限于英语。

那么我们能找到更多的字母幻方吗?答案是肯定的,其理由则简单得不值一提:只要在上述字母幻方中每一项的前面加一个"one million"(100万)就行了,这样,所得到的数字幻方的幻常数增加了300万。而其对应的logorithm幻方—即把每个数字用其logorithm代替后所得的幻方—的幻常数则增加了"one million and"这一词组中所含的字母数的3倍,即增加了$3\times13=39$。因此,字母幻方(3阶)存在着无穷多种推广,Sallows称这类幻方为原先那个幻方的"谐波",认为它们是没有多大意义的变种而不予考虑。

还有没有更有趣的幻方变种呢?19世纪时法国数学家Edouard Lucas设计出了适用于任何3×3幻方的一个公式:



a+b	a−b−c	a+c
a−b+c	a	a+b−c
a−c	a+b+c	a−b

这个幻方的幻常数是3a。无论用什么数值来代替a、b和c,所得结果总是一个幻方,并且任何一个3阶幻方都可通过这种方式导出。注意,穿过这一幻方中部的那一行、列及主副对角线上的三项分别构成了一个算术级数,即相邻各项之差为常数的级数。因此,找出字母幻方的一个合理方法是寻找若干可以构成算术级数的三项数组,使其相应的logorithm序列也构成一个算术级数。

作为初次尝试,把位于幻方中央位置的数定为15,因为我们已经知道存在至少一个这种字母幻方。然后,根据logorithm表,可知有5个相应的三项数组,即(2,15,28),(5,15,25),(8,15,22),(11,15,19)和(12,15,18)。现在我们在幻方的两条对角线上尝试这些三项数组的所有各种两两组合的形式。例如,如果我们用头两个三项数组,就得到下面这种形式:

2		25
	15	
5		28

根据lucas的公式可知,任何一个三阶幻方的幻常数都是中间一项的3倍,因此,任何一个其中间一项为15的幻方,其幻常数都是45。因此,上面这个幻方的所缺各项只能有一种方式填上,即:

2	18	25
38	15	−8
5	12	28

这个例子中出现了—8这一项。我们可以舍去这个例子不予考虑,也可以把—8写为“minus eight”,这样它的logorithm就是10。

在后一种情况下,我们就得到了下面这个 logorithm 幻方:

3	8	10
11	7	10
4	6	11

不幸的是,这个方阵其实并非幻方。对其他各对三项数组依次进行尝试,很快便可发现一个新的字母幻方。读者们可以把它作为一个绝好的热身问题亲自试一下(答案见本文末)。如果中间一项不是15,情况又如何呢? Sallows 编写了一个计算机程序来搜寻其他的 3 阶幻方,并且发现了不少。一个例子是

15	72	48
78	45	12
42	18	75

也可以用其他语言来玩这个游戏。本文的框内文字给出了斯瓦希里语、威尔士语、法语和德语的例子,Sallows 使用 100 以内的数找到了 19 种语言中的三阶字母幻方—不过没有发现丹麦语和拉丁语有这种幻方。

在法语中,涉及 200 以内的数的字母幻方恰好有一种,但如果把各项的大小增加到 300,则可以又找到 255 种幻方,其中有 3 个幻方的 logorithms 构成了一系列连续的数。本文中给出了一例。

在德语中,使用 100 以内的数的幻方多达 221 个,本文中示出了一例。为了看出构造这些幻方的基本方法,我们把音节"und"和"zig"去掉,并把所得的各个单词用相应的数字来代替,就得到:

$\underline{4}5$	$\underline{6}2$	$\underline{5}8$
$\underline{6}8$	$\underline{5}5$	$\underline{4}2$
$\underline{5}2$	$\underline{4}8$	$\underline{6}5$

我在每项的第一个数字下面划了一根短线,因为这个数字必须



乘以 10 后才能与第二个数字一起构成该项实际数值，例如，“fünf-und-vierzig”意味着“five-and-forty”(45)。现在把这个方阵划分为划线的与未划线的两部分：

4	6	5
6	5	4
5	4	6

5	2	8
8	5	2
2	8	5

每一个都是所谓的拉丁方，即同样的三个数字在每一行和每一列中重复出现的方阵。这样它们的行和列不消说自然具有幻方的性质。此外，凑巧的是，它们的生副对角线也符合幻方的要求。把上面第一个幻方的各项乘以 10 并把两个幻方“相加”后，情况依然如此，由于此方阵中每个数的 logorithm 均相同(14)，因此原始方阵自然属于字母幻方之列。

阶数更高的幻方情况又如何呢？正交拉丁方的技巧是非常管用的。例如，在英语中，下述数字方阵是字母幻方：

26	37	48	59
49	58	27	36
57	46	39	28
38	29	56	47

其下划有短线的数字和未划短线的数字分别构成了一个 4 阶拉丁方，而 20 到 99 之间的英语数词结构的规律则使二者得以结合从而构成这个字母幻方。Sallows 称这类例子是“愚人的金矿”(黄铁矿)，因为它们太容易了，以致难于注意到。为了得到真正的金矿，读者必须寻找一些不一般的例子，如：



31	23	8	15
17	5	21	34
26	38	13	0
3	11	35	28

这个领域中悬而未决的问题的情况如何呢？有3个未解决的问题，读者可以使用任何一种字母语言来对付它们。

1."正规"方阵中使用的数字是从1开始的连续整数。对于3阶方阵，仅存在一种正规幻方(除开通过旋转和反射得到的变形以外)，而这个幻方不是字母幻方。4阶的情况又如何呢？"one，two……sixteen"(1,2……16)这一串数词中的字母的总数为81。因此，其logorithm方阵的幻常数必定为81/4，而这个数字不是整数，因此不可能存在正规的4阶字母幻方，按照这一论证方法，可以证明，英语中如果存在正规的字母幻方的话，其阶数至少应为14，其幻常数必定为189。似乎还没有人知道是否真的存在这种幻方，这是第一个未解决的问题。

2. 是否存在3×3×3的字母幻立方？

3.logorithm方阵是对一个数字方阵中的各项取其logorithm值后得到的另一个数字方阵，因此，重复进行这一操作，可以得到第二个logorithm方阵，如此类推。如果要使每个方阵均为幻方，这一过程可以进行多远？在没有其他条件的情况下，这个问题的答案是可以"无限"地进行下去。欲了解其原因，试考虑一下前面分析过的德语字母幻方，它的logorithm方阵每一项均为14，显然是个不值一提的幻方；而它的二级logorithm方阵的每一项则为8，自然也是个幻方。但是否存在其logorithm方阵的各项并不全都相同的这类"递推幻方"的例子呢？



热身问题的答案

8	19	18
25	15	5
12	11	22

国际字母幻方

以下各例中,括号内的第一项是数字值,第二项则是它的 logorithm 值。

斯瓦希里语

arobaini na tano	sitini na saba	hamsini na tisa
(45,14)	(67,12)	(59,13)
sabini na moja	hamsini na saba	arobaini na tatu
(71,12)	(57,13)	(43,14)
hamsini na tano	arobaini na saba	sitini na tisa
(55,13)	(47,14)	(69,12)

威尔士语

chwech deg dau	wyth deg	saith deg pedwar
(62,12)	(80,7)	(74,14)
wyth deg pedwar	saith deg dau	chwech deg
(84,13)	(72,11)	(60,9)
saith deg	chwech deg pedwar	wyth deg dau
(70,8)	(64,15)	(82,10)

法语

quinze	deux cent six	cent quinze
(15,6)	(206,11)	(115,10)
deux cent douze	cent douze	douze
(212,13)	(112,9)	(12,5)



cent neuf　dix-huit　deux cent neuf

(109,8)　(18,7)　(209,12)

德语

fünfundvierzig　zweiundsechzig　achtundfünfzig

(45,14)　(62,14)　(58,14)

achtundsechzig　fünfundfünfzig　zweiundvierzig

(68,14)　(55,14)　(42,14)

zweiundfünfzig　achtundvierzig　fünfundsechzig

(52,14)　(48,14)　(65,14)

9. 走出迷宫

迷宫在严肃的数学研究中比人们想象的要常见。事实上，任何一项数学研究都要求你找出穿过一个由各种陈述构成的迷宫的路径，从每项陈述通向下一项陈述的路径为一个有效的逻辑推理。佛罗里达州丘辟特市的 Robert Abbott 发明的一类新迷宫——称为“母牛在何处?”——既是几何上的，又是逻辑上的。

此迷宫的基础是一种名为“自指示”的逻辑把戏。自指示语句令逻辑学家和哲学家们大伤脑筋。该类语句的一个例子是与克里特人 Epimenides 有关的一个悖论。Epimenides 宣称所有克里特人都说谎，这一说法可简化为：

本语句是假的。

那么，它究竟是假的还是真的? 无论何种情况都会导致矛盾。还有与之类似的互指示语句：

下一句是真的。

上一句是假的。

这是一个逻辑陷阱。



自指示是逻辑学家的一个重要的研究领域。但是在我看来,真正重要的问题应当是:自指示能用来使迷宫更迷惑人吗?这个问题的答案是肯定的。

Abbott 的迷宫示于下图。不但其中的文字是自指示的,而且迷宫的规则也随你的走法而变。为了穿过迷宫,你需要双手并用。每只手各握一支铅笔或其他某种尖物以提醒你现在何处对你是有帮助的。开始时把一支铅笔放在方框 1 上,另一支铅笔放在方框 7 上(方框的编号不是严格按顺序的,而是经过审慎的考虑)。你的目标是在迷宫中逐步移动,使得最终至少有一支铅笔指向标有"GOAL"(目标)的方框。为了走动一步,需要选择一支铅笔,然后根据这支铅笔所指向的方框中的规定采取行动。就是这么回事。不需要作其他选择,但在按照方框 55 中的规定行动时则是例外。

例如,假定你的第一个选择是选择了指向方框 7 的那支铅笔。这个方框内的问题——"另一支铅笔指向的方框的编号是否为奇数?"——的答案显然为"是"。这样你就必须把方框 7 中的铅笔沿着标有"是"的路径移动,这样就到了方框 26。

易如反掌吧?不过且慢。假定你现在选择的是指向方框 26 的那支铅笔。"如果你选择的是另一支铅笔,那么它是否沿着标有'否'的路径离开?"嗯……先前另一支铅笔指向(现在仍然指向)框 1。如果你选择了这支铅笔,那么问题就变成:"另一支铅笔指向的框内有没有红色的文字或绿色的文字?"方框 7 内确实有红色的文字,因此方框 1 中的问题的答案就是"是"。这样指向方框 1 的铅笔就将沿着"是"的路径离开。所有这一切意味着方框 26 中的问题的答案是"否"。因此,指向方框 26 的铅笔现在就沿着"否"的路径移动,到达方框 55。

噢!

大多数方框都是提出问题,你离开方框的路径与问题的答案有

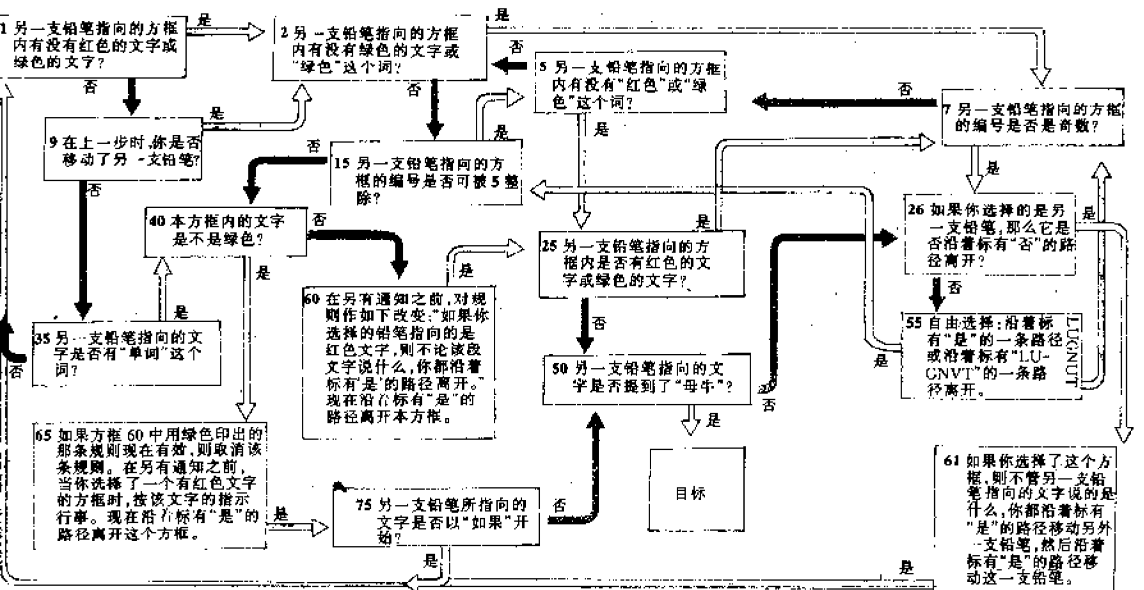


图 “母牛在何处？”是一个自指示的迷宫（实际上它同母牛毫无关系）



关。然而，有些方框的情况不同。方框 61 规定两支铅笔都要移动，只有在两支铅笔都动了之后这一步才算结束。方框 55 有一条标有“LUGNUT”(而不是通常的“否”)的离开路径。这一差别是有作用的——例如，如果你的另一支铅笔指向的是方框 26 的话。

真正惊人的方框是 60 和 65。方框 60 改变了离开有红色文字的方框的规则，代之以这样一条规则：“不论该框文字作何规定，你都沿着标有‘是’的路径离开。”我把这条规则称为规则 60。方框 65 废除了规则 60。有可能出现这种情况，即你的一支铅笔指向方框 60，而另一支铅笔指向方框 65。每个方框都要你别管另一个方框的规定。但是这不会造成自指示问题，因为你必须决定服从哪一个方框的命令；你不能同时服从两者。

某些规定看起来似乎是模棱两可的。例如，方框 5 询问另一支铅笔指向的文字是不是有“红色”或“绿色”这个词。如果另一支铅笔指向的是方框 1，那么其答案很明显为“是”。然而，如果另一支铅笔也是指向方框 5，那么情况又如何呢？方框 5 的“红”与“绿”都放在引号内，这算不算数呢？Abbott 的解释是引号无关紧要，因此答案应为“是”。此外，方框 50 问的是另外一支铅笔所指向的文字中是否提到“母牛”。但是，“母牛”这个词并未出现在其他任何一个框中。当然，两支铅笔可以都指向方框 50，在这种情况下你就可以离开此框而到达“目标”，除非你认为方框 50 不算是提到“母牛”。你应当避免这类哲学上的钻牛角尖，否则你永远不能穿过这迷宫。

现在你可能相信到达目标的唯一途径就是两支铅笔同时指向方框 50 了。如果没有方框 60，那么情况的确如此。如果你得以在规则 60 有效时使一支铅笔指向方框 50，那么不论另一支铅笔指向何处，你都可以到达目标了。事实上，存在着另外一条可以想得出的途径到达目标；你能找到它吗？

可能出现的最奇妙的情况是两支铅笔都指向方框 26。此时该



方框内的问题不存在显而易见的答案。那么会发生什么情况呢？Abbott 巧妙地布置了他的迷宫，使得两支铅笔同时指向方框 26 的情况只有在规则 60 有效时才会出现，这样对方框 26 中的文字说些什么就可以置之不理了！如果两支铅笔同时指向方框 61，情况也是如此。

现在你真正应当做的就是自己来试一下，无须更多的提示了。如果你不希望靠瞎撞乱摸的笨办法来解决问题，你可以使用几种计策。一条计策是找出迷宫的关键特征。例如，为了到达目标，你必须有一支铅笔指向方框 50，同时必须能沿着“是”的路径离开此框。另一条计策是从一个期望的位置出发向后倒推。

享受一下过迷宫的乐趣吧？

提　示

如果你尝试了上面所有方法而依然陷在迷宫中出不来，以下几点提示可能对你有帮助：

1. 为了到达目标，你必须到达(50,50)这个位置，此时两支铅笔都指向方框 50，且规则 60 不生效。另外两种到达目标的方法实际上是不能实现的。

2. 为了到达(50,50)，你首先必须到达(35,35)。此时你离目标尚有 18 步之遥。

3. 为了到达(35,35)，你必须到达(61,75)，并将铅笔移到方框 61。然后两支铅笔都可移到方框 1。从方框 1 就容易到达(35,35)了。

4. 从起点(1,7)到达(61,75)有许多条路径，所有这些路径都要求你使方框 60 中的规则生效，然后又通过方框 65 取消这条规则。

解　答

在下面每一个括号中，下面加了短线的数字表示你应当移动的铅笔。星号表示规则 60 有效。



(1,7),(1,26),(2,26),(15,26),(26,40),(26,60),(55,60),(25,55)*,(7,55)*,(26,55)*,(55,61)*,(15,61)*,(40,61)*,(61,65)*,(61,75),(1,1),(1,9),(1,35),(9,35),(35,35),(35,40),(35,60),(25,35)*,(7,35),(26,35)*,(35,61)*,(1,35)*,(9,35)*,(2,35)*,(15,35)*,(5,35)*,(5,40)*,(25,40)*,(25,65)*,(25,75),(50,75),(50,50),到达目标。

10. 巧破盗窃案

当我走进福尔摩斯的住所时，我发现他正在收拾行囊。“华生，我们必须马上赶到 Ghastleigh Grange 去。Ghastleigh 公爵的处境极其危险。以前在 Grange 当过管家并因犯有谋杀一位女仆的罪行而被判监禁的 Hugh Dunnett 已越狱潜逃了。”

在路上，我仔细阅读了一本关于数学难题的薄薄的书。“福尔摩斯，这里有一道精彩的难题。如果某个人在一个湖的湖心处，而大雾笼罩着湖面，那么他可选择的通往陆地的最短路径是哪条？很明显无人知道。不过这本书却认为，最短路径是首先向前直走一会儿，向左急转后再直走一段，绕个弯路后又径直向前。”

“真是妙极了”，福尔摩斯说，几乎一点不掩饰他的嘲讽口气。“瞧，我们已经到了”。

公爵看起来憔悴不堪。“福尔摩斯，我担心 Ghastleigh 山羊已经被偷走了。”他所说的山羊是家族的传家宝，用青铜制成，约 3 英尺长。它实际上不值多少钱，但却藏有一个秘密抽屉，其中装满了重要文件。



"Dunnett",福尔摩斯咕哝着说,"快带我们去看看你保藏这只青铜山羊的地方。"

公爵领着我们来到一个小小的通风地下室。"就在那里",他指着角落里的一个大保险箱说道。福尔摩斯仔细查看了下水沟,盯了盯通风铁条并检查了地下室门上的锁和保险箱的锁。他跪下察看时发现了一张包装纸。然后他闻了闻空气的气味,站了起来。

"情况很清楚,大人。窃贼是从通风铁条进出的。他打开了保险箱上的锁,拿走了青铜山羊。山羊不能穿过通风铁条,因此他把山羊拴在一个可充气的橡皮管上,扔进下水沟里漂浮出来,他再在庄园外面捡回山羊。"

"不过,这橡皮管上必定有一个洞。山羊还没有来得及漂浮出去就沉到沟底,堵塞了下水沟,你可以闻得出来文件就在下水沟里。但是我们从这一头无法搞清楚它们的位置。这下水沟太深了。我们必须找一个更合适的地方进入下水道中。"

"有一条下水沟穿过前面的草地后进入地下室",公爵提出,"夏天可以看出这下水沟在哪里,因为沟上面的草的颜色不同。但现在它盖满了雪。如果我回忆得不错的话。这下水沟是在距水神雕像100码以内的地方穿过的。"

"我们必须挖一条沟通到这下水沟,确定它的出口并抢在Dunnett之前找回山羊。"

"我们得尽快动手挖沟!"公爵催促道。"还得顺着正确的方向挖",我插上一句,"否则我们有可能根本找不着下水沟。"

福尔摩斯说:"现在我们需要知道的是肯定能同距水神雕像100码以内穿过的所有直线相交的最短沟道。"(见图1)

"我们可以挖一条半径为100码的圆形沟",公爵提议。

"这样周长就是200π码(即628码)"福尔摩斯迅速地算了出来,"能不能简短一点呢?"



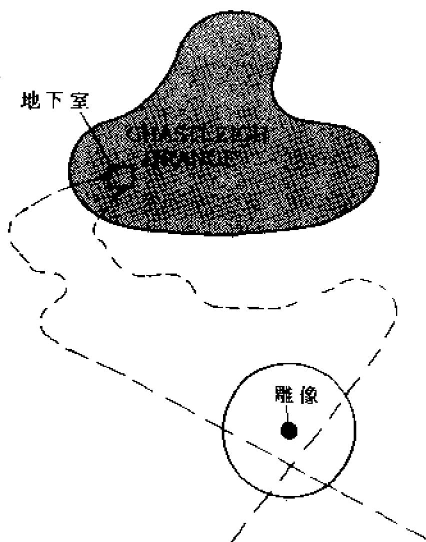


图1　此图示出了穿过有标志的圆的下水沟可能的路径

“挖一条200码长的直线穿过公爵所说的那个圆又如何呢?”我问。

“不错,华生。只不过这样一条沟将会漏掉下水沟的许多可能的位置。”

“噢。那么挖两条这样的直线又如何呢?这两条线相交成直角,总长为400码。”

“还是这个问题,华生。从数学上说,我们是在寻找与半径r为100码的圆的每一条弦都相交的最短曲线,而弦则是与圆有两个交点的任意一条线段。”

“福尔摩斯,想想看有多少条弦需要考虑!”

“是的。如果能简化这个问题就好办了。哦,有了。事实上,我们只需要考虑这个圆的切线,华生。这些线与圆在其边缘的一点上



相交。无论如何,与一个圆的所有切线相交的曲线必定也与所有的弦相交。"

"选择任一条弦,并考虑与这条弦平行的两条线(见图2)。那一曲线与一条切线交于点B,与另一曲线交于点C。由于曲线是连续的,因此位于点B与点C之间的那一段曲线必定穿过弦。"他揉了揉下巴。"我几乎快要解决这个问题了,可是我的推理中仍有一个空白。"

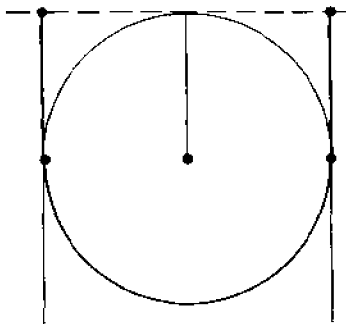


图2　与下水沟相交的最短的沟由一个半圆(BC)和两条切线(AB和CD)构成

"我们没有时间去为细节操心了,福尔摩斯。快谈谈你总的想法吧!"

"嗯,有一类曲线是自动地与所有的切线相交的,这就是其起点和终点分别位于同一切线的相对两侧,包着圆绕一圈的那些曲线。它们始终保持在圆外或在圆上。"我把它们叫做带子,因为它们把一条切线系在了圆上。"

"说下去,说下去。"

"找出最短的一条带子是很简单的,首先要注意到,这条带子必须与圆在某点相接触。否则我们就可以把它收紧一点,直至它与圆接触,这样它就会短一些了。假定这带子与圆最先接触的地方是点



B,而最后与圆接触的地方是点 C。那么,AB 与 CD 必为直线,否则就可以通过把 AB 与 CD 拉直来使带子缩短了。此外,由于类似的理由,BC 必定为圆的一段弧。"

"我认为 AB 与 CD 必定是圆的切线",我说,"如果它们不是圆的切线,那么就可以把 B 点和 C 点移到使 AB 和 CD 与圆相切的位置上,从而进一步缩短曲线的长度。"

"当然如此。但 A 和 D 这两点的位置应在何处呢? 我认为 AB 和 CD 这两条线都必须与切线 AD 相垂直,否则我们就可以使带子来回摆动以使 AB 和 CD 与切线 AD 相垂直,这样又把带子缩短了。"

"不错",我叫了起来,"弧 BC 是个半圆。我们找到了最短的曲线!"

"遗憾的是,我们仅发现了最短的带子",福尔摩斯皱着眉头说,"但是我的确发现很难看出满足我们要求的任何曲线怎么能更短。"

我们一声不吭地站了几分钟。"并非一无所获",我最后说,"这条最短的带子究竟有多长?"

"$(2+\pi)r$,在这个例子中就是 514 码。"

"那比起我的计划来短了 89 码",公爵叫道,"没有时间可浪费了。我马上召集人!"

当他们在挖沟时,我和福尔摩斯仍在继续搜寻更短的曲线,但却一无所获。"福尔摩斯! 你还记得我的书吗?"我把书从口袋里掏出来。"听着,'与一个半径为 r 的圆的每条弦均相交的最短路径由两条长度为 r 的平行的直线段和一个半圆构成。'"

"正像我推导的那样。华生,我承认我低估了你那本小书的用处。出于好奇,我想知道,我们怎么肯定所找到的这条路径的确是最短的呢?"

"这一点已由好几位数学家不容置疑地确证了。不过这些证明非常复杂。如果有人能找到一种较简短的证明,无疑是非常有意义

的。”

“或许我能——”福尔摩斯刚开始说，公爵突然发出了一声尖叫。他的挖掘工们找到了下水沟。

福尔摩斯估量了一下下水沟的长度。“我们将在远处的那个灌木丛再过去一点的地方找到出口——以及窃贼。”

我们在树林里藏了起来，并安下心等待好戏开场。太阳刚落山，我们便听到了脚步声。一个戴面具的人出现了，福尔摩斯抓住了他。“现在我们来看看”，福尔摩斯宣布道，“正如我在开始时推理的那样，这位是——你是谁？”

“天哪，这是女仆 Lucinda”，公爵说，“你在这儿干什么？”

“对不起，大人。昨天我有事非得去地下室一趟不可。地下室的门锁着，于是我从通风铁条处爬了进去。保险箱开着，我看见一只很逗人的老山羊在保险箱里。我把它拉出来以便看一下，但它实在太重了，我一不小心让它掉进了下水沟里。我害怕极了。于是我关上保险箱就溜之大吉。我打算把它送回原处——反正，我正要爬进下水沟去寻找它时这位先生扑在了我身上。”她边说边向福尔摩斯丢去一个媚人的微笑。

“这么说来 Ghastleigh 山羊是在下水沟底”，公爵沉思道，“我的草地上挖出了一条 500 码长的沟，而 Dunnett 依然逍遥法外。”他狠狠地瞪了福尔摩斯一眼。

“这是一个逻辑推理的问题”，福尔摩斯说，“当你排除了不可能发生的情况后，那么留下来的无论其可能性多么小——”

“对，福尔摩斯，对！说下去！”我催促他。

“——依然是不大可能发生的”，他接着把话说完。“不过别引用我的话。”

“我会守口如瓶的。”但我的笔记本却不会。不管怎么说，传记作者还得想方设想混饭吃。

11. 分形雕塑作品

Alan St. George是一位英国出生的退休建筑师，住在葡萄牙，从事数学雕塑的创作。St. George的众多创作主题中，有一个是利用规则立体的分形或螺旋变形。虽然他的原作是用丙烯酸塑料板或金属制作的，但其中大多数也可以仅用纸板或木块来复制——对于那些想要使自己的雕塑作品虚拟化的人，也可以用计算机制图技术来复现。基本的原理可以随意加以利用，创作出一些独特的变形来。

欧几里得在他的《几何原本》一书中证明了其各项均为正多边形，且在每一顶点处这些正多边形的排布方式完全相同的立体正好有5种。这5种立体就是正方体、正四面体、正八面体、正十二面体和正二十面体，它们分别由6个正方形、4个正三角形、8个正三角形、12个正五边形和20个正三角形构成。为了作出分形的雕塑作品，St. George以上面五种立体中的一个为出发点，在它的各个面上粘上一系列越来越小的立体，使得其结果看起来像一个不同的规则立体。

例如，为了把一个立方体转变为一个分形的正八面体，开始时从

一个立方体着手,把它的每一面分成9个相等的正方形,这样它看起来就像一个Rubik魔方。然后,用6个较小的立方体做出一个十字形的立体,这些较小的立方体的每一面同上述较大立方体每一面划分成的小正方形相同。5个小立方体排列成正十字的形状,第6个小立方体则放在十字中央那个小立方体的上面,构成一种阶梯状金字塔的形状。在原先那个大立方体的6个面中的每一个上粘一个这种立体,然后把所得到的结构上的每一个正方形面又划分为9个更小的正方形,再粘上更小的阶梯状金字塔立体。

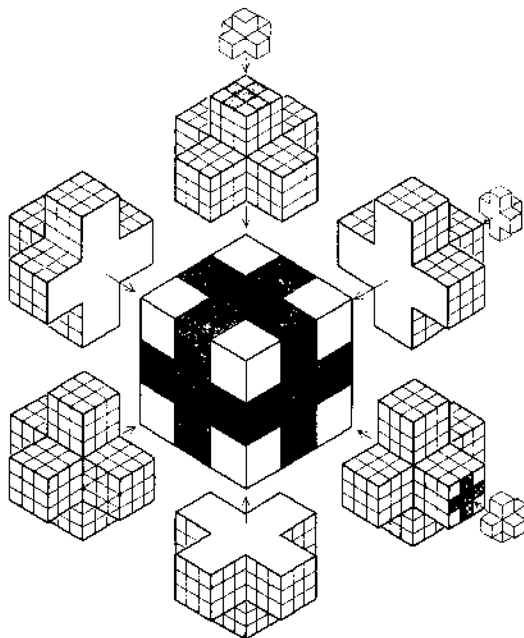


图1 通过分形化步骤在一个立方体上添加较小的立方体,可以得出一个正八面体

到此你可能想停下不干了。最初的立方体有6个面,因此第一



步需要36个较小的立方体构成6个金字塔形立体。每个立体又产生21个新的正方形面，同时留下原先那个立方体每一面划分出的4个正方形未被盖住。因此，原先那个正方体的每一面就变成了25个较小的正方形面。所得的立体有25×6=150个正方形面。下一步似乎必须粘上6×150=900个更小的微形立方体。但是有些立方体是相互重叠的，根据分形方案，只需要粘上708个小立方体。

也可以通过分形化的过程把一个正八面体变成一个正四面体。在这种情况下，把最初的正八面体的一个三角形面划分为四个相同的三角形。并把一个尺寸小一半的正八面体每隔一个面粘在中央的三角形上。这一过程反复进行若干次。如果把那个小的正八面体粘在每一面上而不是每隔一个面粘一次，就可以得到一个立方体。事实上，St. George的雕塑作品把正四面体、立方体和正八面体这些立体中的每一个通过分形化转变成那3个立体中的另外任一个。

涉及正十二面体的分形化过程比较麻烦，因为一个正五边形的面不能划分为若干更小的正五边形。上述过程可以用于正二十面体。但是，所得的结果不像其他的规则立体。这个问题也可以追溯到正二十面体的五边形几何特性，因为在正二十面体中每个顶点周围有5个面。

不过，这一领域还大有可探索的地方。可以把上述设想推广到"半规则"的立体上。所谓半规则的立体就是其各个面包含几种不同类型的正多边形的立体。这类雕塑中的大多数的最简便的制作方法是用低板组装出组分多面体，再把这些多面体粘结在一起。一个简单的"模板"，例如在若干适当位置上有些针孔的一块硬低板，可以保证相同的亚单位立体真正是相同的。

St. George的螺旋形依据的是同样简单的几何原理，但它们的制作过程更复杂，需要通过试错法才能完成。下面我将介绍以正二十面体为基础的最复杂的螺旋形中的一种。

这一制作过程依靠一条闭合的、几何形状规则的回路，它沿着正二十面体的棱延伸，且只通过每个顶点一次。想象有一只蚂蚁把这个正二十面体的各棱当作一个公路网。蚂蚁在爬近一个新的顶点时，可以选择在这个顶点相交的 5 条道路中的任何一条。它可以倒转方向，沿着同一条棱返回，也可以向右急转，向右缓转，向左缓转或向左急转。把后面这四种选择分别记为 R，r，l，L（我们不需要第一种选择）。从标有 A 的一点出发向上爬，蚂蚁循着一条由规定各次转弯的方向的字母序列 RlLrRlLrRlLr 所形成的回路走下去。注意在这个序列中子序列 RlLr 重复 3 次，呈现出明显的规律。

出于美学上的原因，St. George 使用了有名的“黄金数”$\varphi=(1+\sqrt{5})/2=1.618\,034$ 来制作螺旋形。这个数有许多和谐的特性，如 $1/\varphi=\varphi-1$，$\varphi^2=\varphi+1$ 等，使用这个深受希腊人喜爱的数，就可以获得比例匀称令人赏心悦目的结构。

螺旋形的正二十面体是用相应于蚂蚁旅程中爬过的 12 条棱的一系列“腿”制作出来的。每条腿与先前的一条腿相连接，并与一条棱平行。但是相继的腿的长度不同，每一条腿均为其前面一条的 $\varphi^{1/12}=1.040\,916$ 倍那么长。为什么选择这样一个希奇古怪的数字呢？这是因为，在把 12 条棱都加到一条给定的棱上后，最后一条棱将与原先那条棱平行，并且其尺寸增长了 $(\varphi^{1/12})^{12}=\varphi$ 这样一个倍数。

为了作出这样一种螺旋形的模型，首先应当计算出 $\varphi^{1/12}$ 的各次幂的表，以便能找出相继各边的长度。该表中头 25 个数（精确到十分位）为：1.00，1.04，1.08，1.13，1.17，1.22，1.27，1.32，1.38，1.43，1.49，1.55，1.62 = φ，1.68，1.75，1.82，1.90，1.98，2.06，2.14，2.23，2.32，2.41，2.52，2.62 = 1 + φ。从比如说 2 英寸（5 厘米）的一条边开始，依次乘以上表中列出的各个数，便得出了各条边的长度。

制作螺旋形的最容易的方法就是从硬纸板作成的长条形入手。



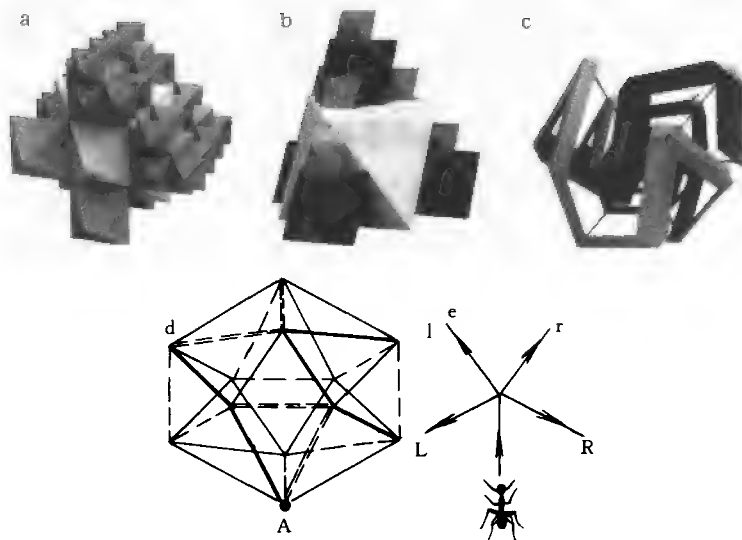


图 2　雕成的立方体(a)和正四面体(b)是 Alan St. George 用一个正八面体作出的。如果有一只蚂蚁在"螺旋 9"(c)上穿越的话(从标有"A"的顶点出发)它就展现出一个正二十面体(d)。蚂蚁沿着一条特殊的路径爬行(用虚线标出),它在每个顶点处所面临的路径选择在(e)中用字母标出

把每一个长条纸板沿中线对折,两端剪成"箭头"形状,在折线的两侧箭头均为 60°的角。在两条腿相交的角上;把箭头的相邻棱粘起来,并填入两个小的等边三角形。

为了使所得的结构是刚性的,还应当保证在每一个角上交会的五块三角形纸板具有与正五棱锥的五个侧面相同的空间排列方式。实现这一点的一种方法是在每一个角上粘一个小型的刚性棱锥(此棱锥带一个正五边形底面)。读者仍然还需要一些支架来撑起这个结构,同时对某些在所难免的不精确之处也需要作一定的调节。如



果这些设计问题得以解决——我将把这个令人愉快的(不过也是冗长的)任务留给诸位读者——那么读者将会得到一个精巧的三维雕塑作品,它必定使任何一个有数学头脑的人的住所增色不少。

12. Monopoly 游戏公平吗

每个人都玩过 Monopoly(垄断)这种游戏。不过我想没有几个人思索过这种游戏包含的数学奥秘。事实上,赢得 Monopoly 的概率可以用一类称为“马尔可夫链”的有趣的数学结构来描述。本文将不考虑他的大部分成果,也不打算考察 Monopoly 的所有规则,但是将使你相信这种游戏是公平的。首先我们应当回顾一下如何玩这个游戏。游戏者轮流掷一对骰子。骰子掷出的点数决定了游戏者可以沿棋盘周围走多少格。掷出一个“双”——比如说两个 1(蛇眼)——的游戏者可再掷一次。所有的游戏者都从标有“GO”的一格开始。

有些点数(如 7)自然比其他点数出现的机会多些。在骰子点数的 36 种可能的组合方式中,有 6 种方式可得到 7(1+6,2+5,3+4,4+3,5+2,6+1)。因此掷出 7 点的概率为 6/36,即 1/6。紧随其后的是 6 点和 8 点,它们出现的概率为 5/36,然后是 5 点和 9 点,其概率为 1/9。再后面是 4 和 10,概率为 1/12。3 和 11 出现的概率为 1/18。最后是 2 点和 12 点,它们出现的概率只有 1/36。从这些数字中我们可以得知,在多次游戏的过程中,第一位游戏者走到第 7 个

格子——即“运气(Chance)格——的可能性最大。如果他没有掷出7点,则他大概可能走到“东方大街”(Oriental Avenue)或“佛蒙特大街”(Vermont Avenue)这两个格子。它们分别位于“运气”的两侧。因此,第一位游戏者有极好的机会把这些地产的某一个弄到手。如果他的确买下了一个,那么就减少了其他游戏者在各自第一次掷骰子时进行购买的机会。

这一事实无疑是 Monopoly 的设计者们把廉价的地产摆在出发点附近的原因之一。昂贵然而可赚大钱的“公园广场”和“百老汇大街”则在棋盘上走几轮后才会到达的地方,到那时推测起来各位游戏者的机会大概已经拉平了。但是否的确如此呢?为了解决这个问题,我将引入另一种简化的方法。我们不考虑一次同时掷两个骰子,而是想象一次掷一个骰子。每个游戏者可以走两步,一个是“虚”步,此时他走到哪里无关紧要,另一个才是实步。类似地,我们将用数学家的眼光来看待棋盘(见彩图1)。

为了方便起见,把棋盘上的格子从0到39一个一个地编号。格子40“返转”到格子0(即GO)上。我们可以认为这些数是按格子40计数的——也就是说,任何一个大于39的数都可以用它除以40后所得的余数来代替。现在想象只有一个游戏者在反复地掷一个骰子,并按掷出的结果在棋盘上走动。在掷了一定次数后,他走到某一给定方格的概率有多大呢?我们希望的是,掷的次数越多,走到棋盘上40个格子中任何一格的概率就越接近1/40。换言之,走到任一格的可能性将变得相等。

求出这些概率的方法就是观察它们的分布是如何随时间“流动”的。每一个分布都可以用40个数组成的一个序列来表示,每个数分别给出了到达每个格子的概率。当游戏开始时,游戏者在格子0(GO),其概率为1(或者总是为1)。因此,其概率分布为1后面跟39个零。经过一次虚掷骰子后,分布变成了0,1/6,1/6,1/6,1/6,



1/6,1/6,0……0——也就是说,走到头6个格子中的某一个的概率为1/6,而且游戏者不可能走到其他任何一格上。

应当注意,总概率(即1)开始时是完全集中在格子0上,现在它则分成了6个相等的部分,分布到其后的格子1到格子6上。这一过程具有普遍性。每当掷了一次骰子之后,一个格子上的概率便除以6。这6个相等的部分沿着顺时针方向流到其后的6个格子中的每一个上。这样,在下次掷骰子后,格子1上的概率(1/6)便重新分布如下:0,0,1/36,1/36,1/36,1/36,1/36,1/36,0……0。格子2到格子6上的1/6也以类似的方法重新分布,但依次向前移一格。

最后,我们把每个特殊格子上已得到的概率相加。例如,格子6从前5个序列中的每一个得到1/36,而从最后一个序列得到0,因此总和为5/36。最终结果是0,0,1/36,2/36,3/36,4/36,5/36,4/36,3/36,2/36,1/36,0……0。注意这一分布与我们早先对一次掷两个骰子所作的预期是相同的。但是现在我们可以继续进行下去了。在第三次掷骰子(虚掷)时,我们把新序列中的每一项乘以1/6,然后依次把它向前移1项、2项、3项、4项、5项和6项。接着我们把每个方格所得到的概率数字加起来。

很容易编写一个计算机程序来一个接一个地计算这些概率分布。其结果见彩图2,其中第一个分布是在第二次掷骰子时得到的"三角形"分布。随后每次掷骰子得到的概率曲线依次向前移一步。你可以注意到,每掷一次骰子,概率峰值就向右边移几格(事实上,平均说来是移了3.5格,即1,2,3,4,5,6这几个数的平均值)。如果你继续进行计算机模拟,你将会发现,三角形最终将变平,所有的数值变得大致相等。但是为什么此模拟会服从这种模式呢?

为了作出解释,我们需要马尔可夫理论,该理论提供了一种系统的方法来跟踪概率流。此方法的第一步是写出第一个数字的所谓转移矩阵(称为M)。此矩阵是一个有40行、40列的方形表,每一行和

列分别从 0 到 39 加以编号。表中位于 r 行 C 列交点处的一项是经过一步后从格子 r 走到格子 C 的概率。如果 C = r + 1, r + 2⋯r + 6,(模 40),则该项的值为 1/6,否则为 0。然后用 M 进行一次比较专门的计算(见框内文字)。结果表明,经过多次掷骰子后,对任意一个格子,概率的确都趋近于 1/40。这样,借助于马尔可夫提供的一点小小的帮助,我们就能够证明,像 Monopoly 之类的复杂游戏在下面这种意义上是公平的,即从长远来看,走到任一格子的机会既不更大一些,也不更小一些。当然,先走的游戏者还是有点微弱的优势,不过,由于他或她的银行余额有限,这点优势也就算不得什么了。

马尔可夫矩阵魔术

设 M 为转移矩阵。首先计算出 M 的本征值,即包括 40 个数的一组数。数 m 是转移矩阵 M 的一个本征值,如果你能够在矩阵的 40 个顶点上写出 40 个数,使得你把每一个数分成 6 个并让它沿着从该顶点出发的 6 条顺时针方向线流动时,所得的数恰好等于 M 乘以你开始时写下的 40 个数(唷,太麻烦了!用符号来表示更容易一些:对于某个 V,如果有 MV = mV,则 m 为矩阵 M 的一个本征值)。但是有一点值得注意:本征值不一定是 0 到 1 之间的实数。它们也可以是用 $i=\sqrt{-1}$ 表示的复数。

由这 40 个数值组成的一个序列称为本征矢量。现在你要做的就是从你已经计算出的 40 个本征值中找出最大的一个本征值。然后,在对相应的本征矢量作“归一化”处理后,就可以用这个本征矢量按你所希望的近似程度来逼近概率分布了。所谓“归一化”就是使本征矢量的各项相加为 1,如像真正的概率值那样(这一步只是意味着你需要把每一项除以总和)。

由于旋转对称性,实际上不难求出本征值和本征矢量。其中一个本征矢量的所有 40 项均等于 1/40。那么它的本征值是什么呢?嗯,假定你就从这个概率分布开始,把每一个 1/40 分为 6 个相等的



部分(每部分为 1/240),然后使它们沿着各自的 6 条顺时针方向线移动。每个顶点都得到恰好 6 个数,分别来自其前面的 6 个顶点中的每一个顶点,因此其最终结果为 6×1/240,即 1/40。这正是本征矢量所应当做的。在这种情况下,本征值为 1。这里我不打算把其他 39 个本征值告诉读者,它们的表达式相当漂亮(或许只有数学家觉得漂亮)。事实上,第二大的一个本征值的绝对值为 0.964。因此,1 是最大的本征值,而它所对应的本征矢量的确代表了概率分布的长期状态。

13. 蠕虫的毯子

所谓"蠕虫的毯子"这个问题是要找出能够覆盖任何一条长度为一个单位的曲线的最小二维区域。此问题的名称似乎很奇怪，不过你可以把这个区域想象成一块毯子，而把曲线想象成一条蛰伏的蠕虫。它的母亲需要一个我们称之为"万用毯子"的东西——不管蠕虫宝宝蛰伏的姿势如何，这块毯子都可以把它盖住。许多年前，加拿大艾伯塔大学的数学家 Leo Moser 提出了一个令人困惑的，而且至今尚未解决的问题：面积最小的万用毯子具有什么样的形状？

这个一般性的问题的一个显而易见的答案是：此毯子应为一个半径为单位长度的圆盘，因此其总面积应为 π，即 3.141。如果蠕虫宝宝的头枕在圆心上，那么它的尾巴最多也只能搁在一个单位远的地方。1973 年，Emporia 国立大学的 Jnametk Gerriets 和 George D. Poole 描述了一个五边形的毯子，它的面积只有 0.286 个单位。直到不久前，尚无人发现面积更小的毯子。但是，后来 Darid Reynolds 证明了一块面积为$(4+\sqrt{3})/24$(即 0.239)的四边形毯子是万用毯子。在这一数学领域中，Reynolds 的结果是富有戏剧性的。他的方法的



某一变形形式可以得出面积更小的毯子。

为了解释 Reynolds 的方法，我首先将用此法证明一个直径为单位长度的半圆——其面积为 $\pi/8$，或 0.392——是万用毯子。给定蠕虫宝宝的某一种姿势，画一条直线把它的头和尾连接起来(如果蠕虫宝宝弯曲成了一个闭合的环形，则随意画一条线穿过其身体)。现在找出包住蠕虫宝宝的最小矩形；此矩形的两条边应当与刚才画的第一条线平行(见图 1)。将此矩形称为一个"块"。此块的宽度 W 在 0 与 1 之间。对于任意一个 W，我们可以估计出高度。我们知道，当蠕虫宝宝取等腰三角形的姿势睡觉时，任何一个块的最大高度应当为 $1/2\sqrt{1-W^2}$。

这样我们就找出了一系列的块(每一宽度 W 一个)，它们合起来是万用的。无论蠕虫宝宝怎样蠕动，这一系列的块中至少有一个可以把它盖住。我们把这些块叠起来，就可以做出一个万用毯子。Reynolds 称这一结构是蠕虫宝宝的"拼合被子"(Patchwork Quilt)，不过它更像一个镶合物：各块的边缘不需要互相连接，事实上，交叠的部分越多越好。不论怎样布置这些块，它们所盖住的区域肯定是万用的。要弄清其原因，想象蠕虫宝宝将身体弯曲起来。找出能盖住它的那个块。拼合被子包含了那个块以及其他一些东西；因此可以适当布置拼合被子以使选定的那个块能够盖住蠕虫宝宝。

我们的下一项任务是把这些块叠起来，以得出面积最小的被子。如果把这些块对称地排列，使其底边对齐，则它们可以覆盖一个直径为单位长度的半圆。这样，正如先前已指出的，一个半圆的确是万用的。但是并无什么特殊的理由非要选择这种布局不可，而且这些块也不一定非得是矩形的。此外，我们也并不是真正需要上面所使用的从 0 到 1 的所有宽度值。当 W 等于$\sqrt{5}/5$ 时，块就变成了正方形的。任何一个其高度大于$\sqrt{5}/5$ 的块都可以转动 90 度，这样它的高度就小于$\sqrt{5}/5$ 了。将这些块从镶合的半圆去掉，半圆的顶部就少了

一部分,这样它的面积缩小了,但仍然保持了其万用性。

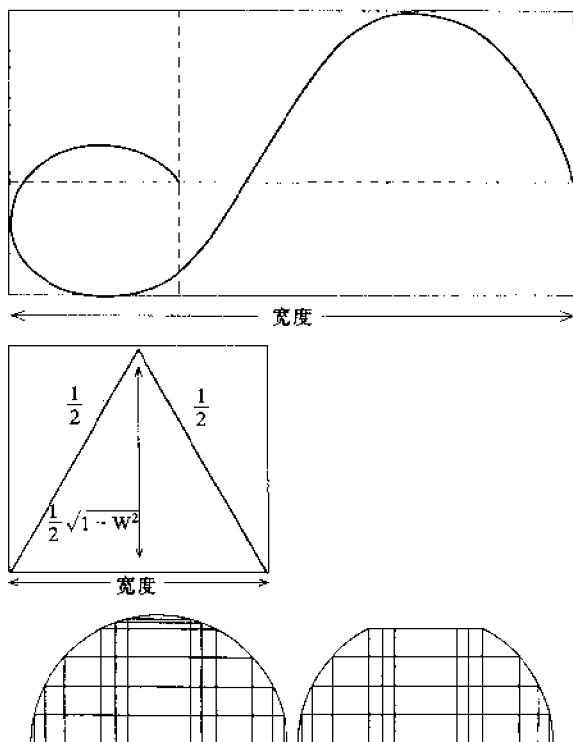


图 1 可以建造出矩形块来覆盖任何一条单位长度的曲线(上图)。当曲线形成一个等腰三角形的两腰时,需要最高的矩形块来覆盖它(中图)。所有这些矩形块经适当布置后可以充满一个半圆形区域(下左图)。对其顶部进行修整后得出一个面积更小的万用毯子(下右图)

通过一种稍微复杂一点的论证——此论证需要转动蠕虫宝宝自身——Reynolds 得出结论:事实上,所有其 W 小于 1/2 的块均可忽



略不计。此外,块仅在其中点需要达到最大高度。除中点以外,其他地方的高度由一段椭圆弓形确定。考虑了这些改进之后,就得出了图 2(a)所示的毯子。初看起来,我们似乎已经挖掘了这一方法的全部潜力,但请注意,毯子还可以翻转过来。如果蠕虫宝宝的身体达到了块的椭圆边界——比如说中心左边的椭圆边界——则它不能也达到中心右边的上部边缘。这样,我们可以去掉每个块的右侧上部边缘的一部分。把这样得到的块重叠起来,就产生一块由一段椭圆形曲线和三条直线围住的毯子。

在构造这些块时,我们假定蠕虫宝宝的头部和尾部是固定不动的。边界的椭圆形部分仅出自于一个块,即宽度为 1/2 的块。但是此块内的任何蠕虫都可以被推向地毯的右端,因为右端还有许多额外的空间。在对蠕虫的若干种关键形状进行推敲之后,Reynolds 证明了我们先前得出的毯子的随圆边缘和左边的垂直边缘可以用两条直线代替,这样就把毯子面积缩小到前面提过的 0.239。

在研究这类问题——即几何优化问题——时,应当记住,它们可能不存在一个最终的解。即使你找到了一系列的解,其中每一个均比前一个有所改进,这一系列的解可能也不收敛。Richard Courant 和 Herbert E. Robbins 在他们的经典著作"什么是数学?"中对我想说的意思给出了一个很好的实例。这个例子就是我所谓的"蚊虫母亲的帐篷"的问题。蚊虫宝宝在睡觉时,一条腿悬在平的地板的上方。蚊虫母亲用一块毯子盖住它的宝宝,这块毯子像帐篷一样,其所有各边都与地板接触。哪种帐篷的表面积最小?

可以使用底面为圆的锥形帐篷,而且底面越小,表面积也越小。事实上,你可以找到其面积为大于零的任何数的符合要求的帐篷。但是,不断改进这一系列的解最终将得出一个面积为零的"最佳"帐篷,而这是不可能的。底面为零的锥不是面,而是一条线段。用一条线段来盖住蚊虫宝宝是没有什么意义的。因此,对于蚊虫母亲的帐

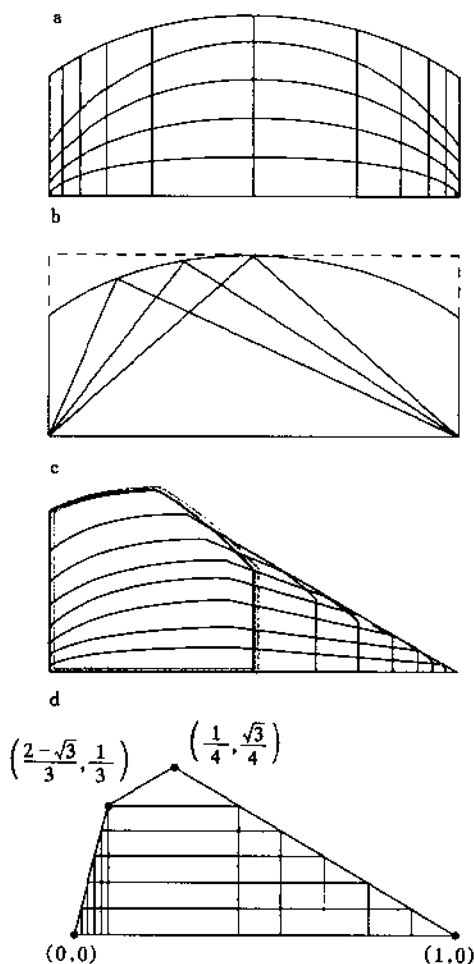


图 2　利用其一边为椭圆形的块(b)，可以作出面积更小的毯子(a)。考虑一下蠕虫宝宝在毯子下的移动方式，能够使毯子的面积进一步缩小(c)。现在已知的最小的毯子，其面积约为 0.239(d)

篷问题，如果我们始终坚持帐篷应当是一个面，那么这个问题就没有最优解。



而且,蠕虫母亲的毯子问题有确定的解这一点也不是很清楚的。在某种程度上,它与我们考虑的是何种类型的毯子有关。例如,所有长度为1个单位的多边形蠕虫都可以用一块面积为零的毯子把它们盖起来。存在许多这类毯子,但是它们基本上都是孔;这个问题最好可称为“鳝鱼母亲的网”问题。另外,英国布里斯托尔大学的 John M. Marstrand 在 1979 年证明了没有任何面积为零的毯子能够盖住每一条平滑的蠕虫(这里所谓的“平滑”,指的是曲线的每一点上均存在切线)。Marstrand 的结果表明,蠕虫母亲的地毯问题的确存在一个相当确定的解,也是一个令人感兴趣的解。Reynolds 的解很可能正是它。

但是业余数学家们仍然可以兴致勃勃地努力寻求更好的解。此外,这个问题还有不少变种,它们几乎全都未得到解决。例如,哪种万用蠕虫毯子的周长为最小?蛇宝宝的睡袋问题又如何(此睡袋必须容纳下一条单位长度的蛇,而无论它在三维方向上如何弯曲盘绕)?对于住在球的表面上的蠕虫,情况又怎样呢?

14. 古怪的雹数

进三步退两步，这不是旅游的最有效的方法，不过似乎可以肯定它能让你到达目的地。数论中有一个有趣的尚未解决的问题却对这个结论提出了疑问。这个问题可以陈述如下：选择任意一个正整数(即任何大于零的整数)，把它叫做 N。如果 N 是奇数，那就把它乘以 3 后再加 1，换句话说把 N 换成 3N+1。如果这数是偶数，那就把它除以 2，也就是把 N 换成 N/2。不管哪种情形，结果都得出一个新的 N 值，然后再对这个新的 N 值重复上述步骤。在多次重复之后，这个数是越变越大还是越变越小？这串数是收敛到某个特殊值还是发散趋于无穷大？要决定一个数的"命运"又需要多少步？

对于任何给定的值 N，回答这些问题只不过需要简单的算术。例如，假如 N 是 27，它是奇数，下一个数值就是$(3\times27)+1=82$；接着是 41，再下去就是 124。显然这串数的值将会不停地上下摆动；因为每当 N 是奇数时，它的值就上升，而 N 是偶数时就下降。读者最好把这个数列延伸下去看看究竟它导向哪里？

对给定的数 N 求出这个数列并不难，困难之处在于对于所有可



能的 N 值,求出一个一般的解答。至今还没有得到一般的解答。但已经对许许多多的数做过明显的试验,结果发现它们全都遵循相同的范式,可是至今还没有一个人能证明每个数的数列都符合这个范式。很难讲这个问题是数论里没有解决的问题中的最重要的一个,可是它肯定是最令人头痛的问题之一。这个步骤很容易讲述,也容易施行,但是要理解计算中所出现的现象却是非常困难的。

这个问题很好地说明了数字计算机作为数学工具的用处及其不足之处。除了最小的那些整数之外,研究这个问题需要有机械辅助工具帮助计算。几乎任何计算机都能干这事,即使是一台可编程序的计算器。另外,当计算已推进到大得多的数的范围中时,实际上只有最强的大的计算机才能办到。至于那些最深刻的问题,肯定任何计算机都无能为力。在很大程度上,计算机是"实验"数学的工具,它可以造出例子和反例。而要在 N 的漫游当中发现一条原理。这似乎要求我们去证明定理而不是摆弄数串。

要是对任意数重复地应用这个变换规则,你能够期望得到什么样的结果?这里有三个初步的猜想:

第一个猜想是这样论证的:奇数和偶数数目相等,因此在任何一长串计算当中,奇数 N 值和偶数 N 值应以相同的频度出现。当 N 为奇数时,它增为 3 倍(还多一点),而当 N 为偶数时,它只减少一半。因此,N 的数值在多次反复变换之后应该无限地增大。平均每步增大为$(3N+1)/2$,对于大的 N 值,实际上就是$3/2N$。

第二个猜想建立在这样一个想法的基础上,也就是有升就必有降。这条思路是从这种观察引起的:每当计算中出现一个 2 的整数幂时,数列就立即像瀑布一样逐级下降到值 1(除了 2 本身之外,任何 2 的方幂被 2 整除的结果必定是一个偶数,因此总是选定计算中的下降的一支)。在无穷多的数目当中有无穷多个 2 的整数幂,因而延续足够长的计算肯定要碰到其中之一。尽管在计算过程中,N 可



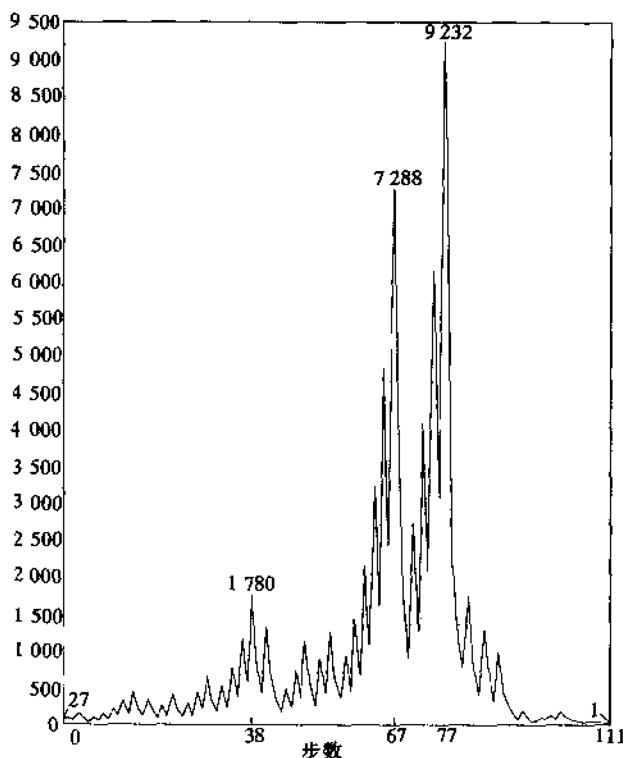


图 1　从 27 开始的雹数序列

以达到极大的数值，可是最终它还是要垮下来。

第三个猜想的论证从形式上类似于第二个，但所得结论不同。注意每当计算改变方向时（比如一串偶数之后碰到了一个奇数），它就又跑到它以前曾经呆过的范围中去。实际上，N 在竖直线上上下地游荡时，能够任意多次地回到一个有限的整数区域中去。可以预料它最后会碰到以前曾经取过的值，一旦发生这种事，计算的整个未来就完全决定下来。因为选取下一步的步骤是被完全决定了的，任何重现的数值必定导致一个无限重复的循环。



这里提出的三个猜想用不着太认真地看待。它们不可能都对。它们的某些前提肯定有问题。特别是,这三个理论都靠的是概率的分析,可是运用这项规则所产生的数列并非随机序列。实验对此会说些什么呢?

我们从1开始计算。1是奇数,从而规则要求把它乘以3再加上1。结果得4,它是偶数,因此除以2,又得到另一个偶数;再除以2就把计算变回到1。因此,这第一次计算就已经对上述三个猜想的理论中的两个给予相当好的支持。正如上述第二个猜想所预言的,计算过程中会碰到2的方幂;而1只走一步之后就碰到2的方幂。正如上述第三个猜想所预言的,计算将会陷入一个无穷无尽地循环里;而在1的情形下,数值4,2,1将无穷无尽地重复下去。

在所有的整数中,1是非常特别的,它是头一个数,也是最小的数。因此当N为1时,所得的结果可能不是典型的;以我们在得到任何结论之前,应该进一步进行检验。因为2和4的命运已经由N=1的计算中得知,从而明显的候选者是3。3是奇数,因此下一个值是(3×3)+1=10。除以2之后得5,然后乘以3再加1得到16。这样就得出2的方幂,从而数列通过N=8递降到4—2—1循环。

表1　N=100 000以下的最长路径序列

N	路径长度	极大值
1	0	1
2	1	2
3	7	16
6	8	16
7	16	52
9	19	52
18	20	52
25	23	88
27	111	9 232

续表

N	路径长度	极大值
54	112	9 232
73	115	9 232
97	118	9 232
129	121	9 232
171	124	9 232
231	127	9 232
313	130	9 232
327	143	9 232
649	144	9 232
703	170	250 504
871	178	190 996
1 161	181	190 996
2 223	182	250 504
2 463	208	250 504
2 919	216	250 504
3 711	237	481 624
6 171	261	975 400
10 971	267	975 400
13 255	275	497 176
17 647	278	11 003 416
23 529	281	11 003 416
26 623	307	106 358 020
34 239	310	18 976 192
35 655	323	41 163 712
52 527	339	106 358 020
77 031	350	21 933 016



表 2　N=100 000 以下的峰值序列

N	路径长度	极大值
1	0	1
2	1	2
3	7	16
7	16	52
15	17	160
27	111	9 232
255	47	13 120
447	97	39 364
639	131	41 524
703	170	250 504
1 819	161	1 276 936
4 255	201	6 810 136
4 591	170	8 153 620
9 663	184	27 114 424
20 895	255	50 143 264
26 623	307	106 358 020
31 911	160	121 012 864
60 975	334	593 279 152
77 671	231	1 570 824 736

在检验头四个自然数之后，其趋势似乎很明显，不过还是有理由对此表示怀疑。到现在为止我们所做的计算中，出现了两个值得注意的数量：一个是 N 在计算过程中可能达到的极大值，一个是路径长度，我们把这个量定义为达到 1 的值所需的总步数。对于 1 本身，极大值为 1，路径长度为 0。对于 2，极大值为 2，路径长度为 1。对于 3，极大值为 16，长度为 7。3 这个例子表明，所能达到的极大值以及



数列的长度可能比 N 的初始值大很多。因此，对于某些 N 值，这两个函数也许会趋于无界。

现在我们再来考虑初始值为 27 时所生成的数列。如上所述，前三个数是 82，41，124，但是连续两次除法又把数列拉回到 31。所以经过五步之后，几乎没有什么进展。不过，随着计算继续下去，这个进三步退两步的机制引起一个具有越来越大振幅的振荡数列。这数列在 142，214，322 和 484 处达到新的高峰。也存在倒退的情形（在第 19 步数值降到 91），可是总的趋势是继续向上。计算先后通过 700，1 186 和 2 158，而且在第 77 步达到 9 232 这个巨大数值。似乎我们得不断这样走下去，不过，结果是在总共走过 111 步之后，数列以 1 结尾，中间再也没有超过 9 232 的数（整个数列如图 1 所示）。

我刚刚进行的这类计算已经对于一个相当大区间内的所有整数都进行过。东京大学的米田信夫已经对 2^{40}（约 1.2×10^{12}）以下的所有整数都进行过试验。在每种情形下，结果完全一样：经过有限多步之后，数列落入 4—2—1 循环中，从而永远地循环下去。在头 50 个整数当中，27 经过最长的路径才到达 1（但 41 和 31 的路径也并不短太多，并也达到同样的峰值，其理由可以由上面的分析明显地看出）。至今还没有发现一个整数，其数列发展下去趋于无穷，也没有发现 4—2—1 以外的任何循环。不过，所有数都符合同样的范式这个猜想还没有任何牢固的理论基础。

这个问题通常称为 3N + 1 问题，它的历史隐晦不清，但似乎并不十分古老。把它称为 3N + 1 问题似乎并不完全合适，因为它不适当地强调了步骤的一半，而忽视了另外一半。在其他各种名称中有一种我觉得是最适宜的，这个名称把从给定初始值所生成的数串称为“雹数”。因为数列所走的路径颇像冰雹穿过暴风云时的轨道，先是气流推着上升，接着又由于自身重量而下降。

用高级程序语言（比如 BASIC）只需几行就可以写下计算雹数的计算机程序。实际上，中心算法可用一条语句来表示。用 BASIC 语言它可写成如下形式：

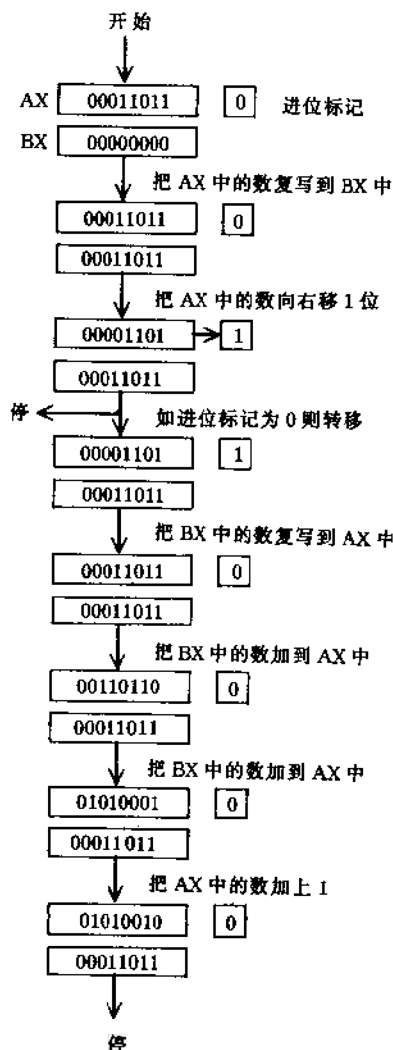

开始
AX
00011011
0
进位标记
BX
00000000
把 AX 中的数复写到 BX 中
00011011
0
00011011
把 AX 中的数向右移 1 位
00001101
1
00011011
停
如进位标记为 0 则转移
00001101
1
00011011
把 BX 中的数复写到 AX 中
00011011
0
00011011
把 BX 中的数加到 AX 中
00110110
0
00011011
把 BX 中的数加到 AX 中
01010001
0
00011011
把 AX 中的数加上 1
01010010
0
00011011
停

图 2　雹数算法



IFNMOD2＝0　THEN　N＝N/2

ELSE　N＝3*N＋1

其中第一个运算，人（而不是计算机）用不着进行明显计算就能做，即判定N是奇数还是偶数。NMOD2是模运算，它算的是N除以2之后的余数。如果余数为0，就执行语句中的THEN部分，将N代之以N/2；否则就执行ELSE部分，即N代之以3N＋1。

用BASIC写的程序对于开头几百个整数生成雹数的计算可以做得相当不错，不过要是更进一步计算下去，它就会慢得让人受不了。BASIC语句要求进行一次除法（作为模运算的一部分）、一次比较，然后或是再一次除法、或是一次乘法和一次加法。除法和乘法都是费时间的运算，在小型计算机系统中尤其如此。这里如果用机器自身的语言直接对中央处理机发出指令，那情况就会大为改进。因为这样所有的乘法和除法运算都可以去掉。

图3是对这种机器语言程序的一个概括叙述。假定N的值一开始在一个寄存器（表示为AX）中，这个寄存器又是进行算术运算的“累加器”。过程开始部分所示的值就是十进位数27的二进制表示。

第一步是把初始值再存入另一个寄存器（标记为BX）中。我们可以利用二进数系的一个性质来避免除法运算，即把二进位数向右移动一位就等价于把这个数除以2，这就像把十进位数向右移动一位就等于用10除一样。在移位过程中，最右端的一位（个位）保存在一个称做进位标记的一位存贮单元之中。通过检查这个进位标记就可以判定原来的数是奇数还是偶数，因为在二进记数法中，奇数的个位为1而偶数的个位为0。

如果N是偶数，那么计算就到此终止。向右移动一位之后在寄存器AX中余下的数值就是商N/2。不过，在这种情形下，要是N是奇数的话就还需要进一步进行计算。首先，从寄存器BX中恢复N的初始值到AX中，然后我们不是把它乘以3，而是把它自己加上自己2次；虽然这样做需要两个机器指令（而不是一个），但是它还是要



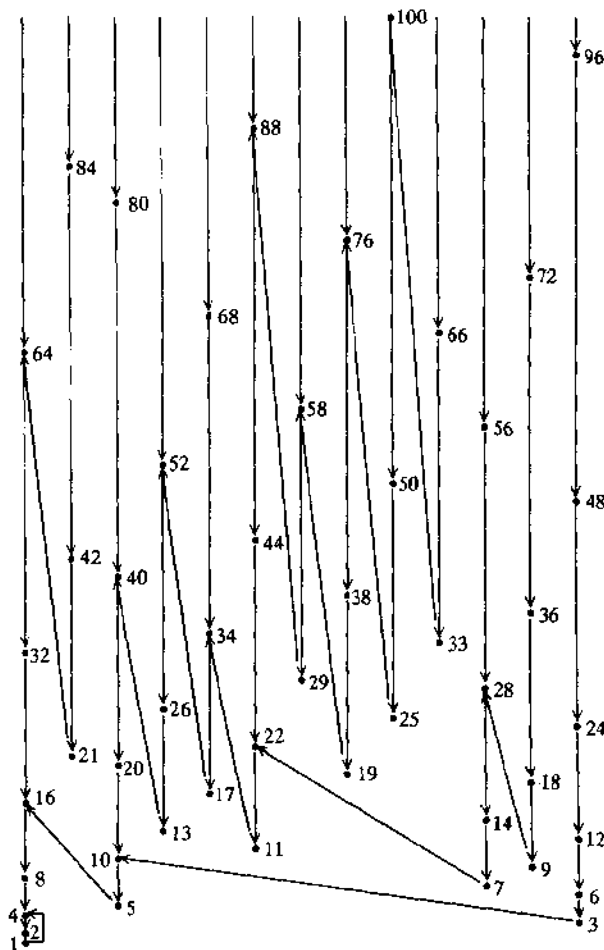


图3　雹数函数反演后所生成的树

快很多。最后一步是把AX中的数加上1。在一个微处理机的指令系统中，整个步骤当N是偶数时需用20个计算机时钟周期，当N是奇数时需用18个周期。如果时钟频率大致是5兆赫的话，原则上这个程序段每秒可被执行250 000次(还可以再节省一些时钟周期，但



这样作程序的清晰性会变差)。而用除法和乘法指令的等价的算法,对于偶数 N 需用 175 周期,而对奇数 N 需用 286 个周期。

在图 3 中所示的寄存器是 8 位宽的,因此能够容纳不大于 2^8 即 256 的数。大多数微处理机中的寄存器实际上是 16 位宽,因此能容纳 65536 以下的数。甚至这个界限也是一个严格的限制;一个使用 16 位算术的程序不能够计算比 702 还大的数的雹数。要想容量更大就要求多精度算术,也就是把一个数分在两个或两个以上的寄存器或存贮单元当中。如精度是 32 位,就可以表示 40 亿以下的数,而 64 位就能把表数的界限推进到 10^{19}。不过,每次精度的增加都要在速度方面付出重大代价。

计算一个 N 值的算法只不过是工作程序的一段。除此之外,还必需有些办法把输入值输入机器并显示结果。而实际考查雹数的程序系统还应该干得更多。例如,它应该能打印出由某一给定初始值所生成的整个数列,还应能列出在某一给定范围中的所有整数所对应的路径长度和极大值。还能设计出另外一种程序来搜索那些产生越来越长的路径或者越来越大的峰值的整数。当然还有其他许多可能性。

在 3N+1 公式中使用不同的系数和常数所产生的各种变形也是值得研究的。R. William Gosper 和 Richard Schroeppel 研究过 3N−1 问题,他们证明这等价于 N 取负值时的 3N+1 问题。他们验证的每个数最后都终止于三个循环中的某一个,其中最长的循环从 N=17 开始,周期为 18 步。

最有效的是这样一个程序,其唯一目标是搜索那些不落入 4—2—1 循环里的数。假如我们从 1 开始相继检验每个整数,则只需要检查奇数就行了。因为任何偶数立刻被减为一半,因此,它所生成的路径先头已经检查过了。出于同样的原因,我们也不必把每个数都一直算到 1 为止;一旦 N 的值降到初始值以下,我们就可以不再考虑它了。

虽然迄今尚未作出关于 3N+1 问题的任何证明,但是从一种较



前述三个初步的假设更为成熟的具有启发性的论证中,可能得出某种解释的线索。前面已经指出,在计算的任何一步上数 N 乘以 3 或除以 2 的概率都是相同的,因而使人想到每步计算后,数 N 可能平均增长到原来的 3/2 倍。但 Lagarias 指出,全体整数中有 1/4 不仅能被 2 整除,而且能被 4 整除;有 1/8 还能被 8 整除,有 1/16 还能被 16 整除,如此类推。如果把这些可用 2 的所有可能的整数幂整除的情况考虑在内,则可得到这样一种估计,即 N 的值平均每步减小到原来的 3/4。实际计算所得的证据是符合这种估计的。

即使结果表明所有正整数最终都进入 4—2—1 循环,雹数仍然给我们提供了大量的有趣事实。这些数最有趣的性质或许是在路径长度及峰值的分布上的明显范型。如果说像 27 这样小的数目的"冰雹"都能在空中上上下下呆上 111 步而且达到 9 232 的高度的话,那么我们可以预料随着 N 的增大,路径长度和峰值将会迅速增大。实际上,路径长度增长十分缓慢,极大值的增长要快一些,可是它还是相当无规律。

在头 100 个整数中,最长的路径是 118 步(N=97);在头100 000 个整数中,最长的路径只不过为 350 步(N=77 031)。因此,N 增加到 1 000 倍,路径长度只增加为 3 倍。这个关系看来是对数关系。在 N=27 处的极大值记录一直保持到 N=255 才被超过,在 N=255 处达到的峰值是 13 120。新的极大值的记录出现在相当不规则的区间上。N=77 671 的雹数序列达到了 1 570 824 736 这个不寻常的高度。

不难看出,在雹数计算中所达到的峰值无一例外都是偶数。我们还能证明,只有奇数 N 值才能打破以前的极大值的记录(N=2 是唯一的例外)。至于创造路径长度的新记录的数,我还不知道是否有理论上的论证要求它们是奇数或是偶数。可是,在头 100 000 整数中,路径长度的记录几乎全是奇数 N 值创造的。

对于一定范围的整数,把它们的路径长度和极大值列成表,你就会发现它们似乎又有规律又没规律,使人难以掌握。它肯定不是随



机数列，不过其范型难以解释。例如，某些极大值比另一些极大值更经常出现，其出现之频繁无法用任何统计过程来解释。一个突出的例子是9 232，这个数首先在N=27场合下达到。在前1 000个整数当中，有350个以上的数以9 232为其极大值。

路径长度的分布也同样的古怪。任何可能的长度都能够产生出来(通过2的相继整数方幂的N值)，可是也存在有的数出现次数比其他数多得多的情况。另外，无论是路径长度还是极大值都表现出成团聚集的强烈趋势。1976年Fred Gruenberger发表了一张这种团的表；最大的团是一串52个连续整数，它们的路径长度完全相同。两个相继的N值是否能具有相同的路径长度和同一极大值呢？这个问题可用代数方法解决，不过喜欢用数字证明的读者通过考察N=386到N=391的雹数序列就可以得到结论。

研究雹数问题还有一个有启发性的方法就是把它上下颠倒过来看。假如所有正整数最终都落入4—2—1循环这一论断是对的，那么它们必定形成一条不断的链条，无穷自然数列中的任何数都通过这条链条连结到底部的循环。这样一来，就可以对雹数函数进行反演，也就是从1开始“倒着”应用变换而生成每一个更大的数。假如用这种办法沿着河流上溯，不能达到某个数，那么这个数就不能以1为其终值。

只要能把这个过程进行到底，这种方法就可能产生出雹数问题的一般解答。不过，结果显示这个过程并不像表面上那么简单。正规的雹数函数是确定的：计算中在任何点的N值只能有唯一可能的相继值。例如，当N为40时，下一个只能是20。如果沿着相反方向走，就会出现多种途径。当我们碰到N=20这个值时，已知它只能由40生成，因此它下一个值必定是40。可是在40处，下一个值可以是80也可以是13。河流分了叉，这样每一支流都必需加以考查。在6K+4型的整数处(K是零或任何正整数)都会出现这种分叉现象。

这类分支系统只能追溯到有限的深度。我们必须沿着某一个分



支追溯到某一个预定的界限，然后把注意转移到另外一个分支上去。如果把界限定在100处，就要考查13个分支，可以证实有49个数与这个由数字组成的河川系统连结在一起。如果界限是1 000，就有84个分支，但是只能数到340个数。如果界限是10 000，就有1 065个分支，它通过4 235个数。注意有一半以上的数似乎是处于河流分支的间隙之中。随着界限的增大，就有更多的数包括进来，但是也有更多的数被遗漏掉了。如果我们能够考查系统的分支到无限的深度，是否所有的整数最后都能在上面有其位置？这是一个尚有待解答的大问题。

15. 你那一半比我这一半大

一位大个子男人和一位小个子男人坐在火车的餐车里，两人都要了鱼。当餐车服务员把鱼端上来后，大个子男人手脚麻利地一把抢运那条大点的鱼；小个子男人于是抱怨说这样太没有礼貌了。

“如果让你先选你会选哪条?”大个子男人生气地反问道。

小个子男人自负地回答:“我当然会表现出绅士风度,选小的那一条。”

“哦,那你不是已经得到你想要的了吗!?”。

这个古老的笑话说明,有些人是很难讨好的。

在过去50年中,数学家们一直在努力研究“公平分配”的问题,它通常是用蛋糕而不是用鱼来表述的。本文将探索一下这个简单得容易使人上当的问题——如何分一块蛋糕使得人人满意——的一些想法。

最简单的例子就是只有两个人分一块蛋糕,使得每人都因分到了公平的一份而感到满意。这时“公平”的意思就是“按我的估价值一半或一半以上”,而分蛋糕者可能对蛋糕的任何一部分的价值有不同的看法。例如,Alice可能喜欢樱桃,而Bob却喜欢糖霜。有趣的



是，当分蛋糕者对蛋糕各部分的价值持不同的看法时，这块蛋糕就更容易分配。

你可以看出本例就正好符合这个道理，因为我们可以把有糖霜的那部分蛋糕分给 Bob，而有樱桃的那部分则分给 Alice，于是两个人都皆大欢喜。如果两人都要糖霜，那么分起来就麻烦一些。

如果只有两个人分蛋糕，倒也不会麻烦到哪里去。"Alice 分，Bob 选"的解决办法现在已经追溯到 2800 年前了！由于这种分法使 Alice 和 Bob 都无权抱怨分的结果，因此它可以称得上是公平的。如果 Alice 不喜欢 Bob 留下的那一份蛋糕，那只能怪她自己为什么没有分得更细心一些以使两份蛋糕完全等值(按她自己的估价)。如果 Bob 不喜欢他的那份，那也只能怨他自己没有选好。

分蛋糕的人数增加到 3 人时，这个问题就开始变得有意思了。Robertson 和 Webb 首先分析了一个似乎正确但实际上却是错误的答案。Tom，Dick 和 Harry 想要分一块蛋糕，使得每人都因至少分得 1/3 感到满意。顺便说一下，我们总是假定这蛋糕是无限可分的(不过，即使蛋糕由可估价的"原子"——即至少有一位分蛋糕者估计其有非零值的单个点——组成，该理论也基本上适用)。这一算法如下：

第 1 步，Tom 把蛋糕分成两份，x 与 w，他认为 x 为整块蛋糕的 1/3，而 w 为 2/3。

第 2 步，Dick 把 w 分成两份，y 与 z，他认为每份均为 w 的 1/2。

第 3 步，Harry 在 x、y 与 z 中任选他看中的一份。然后 Tom 从剩下的两份中选一份。剩下的一份归 Dick。

显然 Harry 将心满意足，因为他是第一个选的。Tom 也将感到满意，其理由稍微复杂一些。如果 Harry 选了 x，那么 Tom 可以选 y 和 z 中他认为更值钱的那一份。由于他认为这两份合起来的价值等于整块蛋糕的 2/3，他必定认为其中至少有一块的价值不低于 1/3。



另外,如果 Harry 选了 y 或 z,那么 Tom 可以选 x。

然而 Dick 可能就不那么高兴了。如果在 Tom 第一次分蛋糕时,他对于两份蛋糕的价值与 Tom 的看法不同,那么他可能认为 w 的价值少于 2/3,这就意味着唯一能使他满意的那份蛋糕是 x。但是 Harry 可能选比如说 y,而 Tom 则选 x,这样 Dick 就只能得到 z 了,尽管他并不喜欢这一份。

这一种公平的分法是 Hugo Steinhaus 在 1944 年发现的(他是定期在利沃夫一家咖啡馆聚会的若干波兰数学家中的一位)。他的方法运用了一种名为“修整”(trimming)的技巧。

第 1 步,Tom 把蛋糕分成两份,x 与 w,他认为 x 为整块蛋糕的 1/3,而 w 为 2/3。

第 2 步,Tom 把 x 递给 Dick,并告诉他说,如果他认为 x 的价值超过 1/3,他可以对其进行修整使其变成 1/3,否则就让这份蛋糕保持原样不动。把这样所得的一份蛋糕称为 x′。可以看出,它或者就是 x,或者比 x 小一些。

第 3 步,Dick 把 x′递给 Harry。后者可以接受 x′,也可以不接受。

第 4 步,(a)如果 Harry 接受了 x′,则 Tom 和 Dick 把该蛋糕剩下的部分——即 w 加上 Dick 从 x 上修整下来的蛋糕,如果他修整了 x 的话——叠成一堆,把它当作单个的一块蛋糕。然后用“我分你选”的办法来分掉它。

(b)如果 Harry 不要 x′,而且 Dick 已经修整过 x,则 Dick 就要 x′,而 Tom 和 Harry 用“你分我选”的办法来分掉剩下的蛋糕。

(c)如果 Harry 不接受 x′,而且 Dick 没有对 x 进行修整,那么 x 就归 Tom,而 Dick 与 Harry 用“你分我选”的办法来分掉剩下的蛋糕。

这就是问题的答案,而我将把它的证明留给读者们去做。概括



地说，就是如果有任何一人对他所得的那份蛋糕感到不满意，那他必定是先前分得不对或者选得不对，这样他就只能怪自己了。

1961 年，Lester E. Dubins 和 Edwin H. Spanier 提出了用一把移动的刀来分蛋糕的方法。让一把刀从左边开始在蛋糕上面平稳地、慢慢地划过。假设 1 是任一瞬间刀的左边那一部分蛋糕。Tom、Dick 和 Harry 被告知，只要他们看到 1 的价值在他们看来达到了 1/3，就立刻大喊“停下！”。第一个喊停下的人分得 1，而剩下的蛋糕则由其他两人用“我分你选”的办法分掉。或者他们也可以让刀在剩下的蛋糕上再次划过，并且在他们看来划出的蛋糕价值达到 1/2 时喊“停下！”(如果有两个人同时喊“停下！”时又该怎么办呢？读者自己去想想)。

这一方法很容易推广到 n 个人分蛋糕的情形。把刀从蛋糕上划过，并告诉每个人说只要他们一看到 1 的价值按他们的估计达到了整块蛋糕的 1/n 时就喊“停下！”。第一个喊“停下！”的人分得 1，剩下的 n－1 个人则对剩下的蛋糕重复上述过程，只是现在他们当然是在看到划出蛋糕的价值达到 1/(n－1) 时就喊停下。如此类推。

我必须指出，我从来不对移动刀的算法感到很满意，因为参与分蛋糕的人的反应存在时间差的问题。最好的办法是慢慢地、非常非常慢地移动刀以避免这个问题。

我们且把第一类解决办法称为固定刀算法，把第二类解决办法称为移动刀算法。有一种三人分蛋糕的固定刀算法也可以不费吹灰之力就推广到 n 个人的情形。开始时是 Tom 一个人呆坐在那里，盯着“他的”蛋糕。接着 Dick 冒出来并要求分得一份。于是 Tom 把蛋糕分成他认为平等的两半，由 Dick 先选一半，剩下一半归 Tom。他们还没来得及尝到蛋糕的美味，Harry 又插了进来，声称他也该得到公平的一份。Tom 和 Dick 就把他们得到的那份蛋糕各自独立地又



分成三份,并认为每一份的价值均相等。Harry 从 Tom 和 Dick 分得的三份蛋糕中各取一份。

不难看出这种依次配对的算法为什么行得通,以及为什么它很容易推广到不管多少人参加分蛋糕的情形。修整法也可以推广到 n 个人的情形,这时只要让围坐在桌旁的每一个人均有机会修整一份蛋糕就行了——如果他们愿意接受其结果的话。

用哪种方法来分蛋糕,切蛋糕的次数最少呢?移动刀方法,只要切 $n-1$ 次就可以把蛋糕分成 n 份,这是切蛋糕所需要的最少次数,不能比这更少了。固定刀算法则要麻烦一些。在 n 个人分蛋糕时,修整算法的一种推广形式需要对蛋糕切 $(n^2-n)/2$ 次,而依次配对的算法则需要切 $n!-1$ 次,其中 $n!$ 为 n 的阶乘,即 $n(n-1)(n-2)\cdots\cdots3\times2\times1$。除了 $n=2$ 这种情形以外,后者所需要的切蛋糕次数比前者要多。

另一种效率较高的固定刀算法是“分割-赢得”算法(divide-and-conquer),其过程大致如下:尽可能一刀就把蛋糕切成这样两份,使得其中一份在差不多一半参加分蛋糕的人看来是公平的而令他们满意,而另一份则在剩下的人看来是公平的而令我们满意。

然后对分开的两份蛋糕各自重复上述过程。这种方法所需要的切蛋糕次数大约为 $n\log2n$。准确的公式为 $nk-2k+1$,其中 k 为满足 $2k-1<n\leqslant2k$ 的唯一整数。这一结果很可能是固定刀算法分蛋糕所能得到的最好结果。

这些分蛋糕的方法原则上也可以应用于某些现实生活的场合。当盟国和俄罗斯出于行政管理的需要而瓜分德国时,第一次瓜分后剩下的地方——柏林——必须再单独地进行划分。这样各方谈判代表就凭直觉运用了分蛋糕的方法。一种相当类似的情况眼下正使以巴关系日益紧张。耶路撒冷是主要的“剩下地盘”,而西岸则是另一块难啃的骨头。



公平分配的数学是否有利于谈判呢？想一下我们若是生活在一个非常讲道理的世界中，以致这样一种分配方式行得通，那该多么好啊！

16. 不引起眼红的分配

我们继续考察与公平地分一块蛋糕这个简单得容易使人上当的问题有关的若干数学问题。所谓公平地分蛋糕的意思就是，如果有n个人分蛋糕，那么每个人都相信自己分得的一份至少为那块蛋糕的1/n。我们将继续研究几个相关的问题，它们涉及到该理论的较现代的内容。

首先我们回忆一下前面所得到的结果。两个人分蛋糕时，使用历史悠久的“我分你选”算法，可以实现公平的分配。当3个人或更多的人分蛋糕时，有几种可能的分法。“修整”法是让参加分蛋糕的人依次把一份据称是分得公平的蛋糕缩小，其条件是如果没有其他人修理这份蛋糕，则最后一个修理者必须挑这份蛋糕。“依次成对”的方法则是让头两位参加分蛋糕的人公平地分蛋糕，而第三人则分别与这两人商量，以确保从已经分开的两块蛋糕中分别取得他或她认为至少有1/3的一份。而在“分割-赢得”方法中，参加分蛋糕的人力求一刀就把蛋糕切成两块，使大约一半的人觉得分到的一块是公平的而感到满意。然后对已经分开的两块蛋糕分别重复同样的方



法,如此类推。

这些算法全都是公平的,但这里还存在一个更复杂的问题。即使人人都相信他或她得到了公平的一份蛋糕,由于妒忌心的作怪,某些人仍然可能觉得自己受了委屈。例如,Tom,Dick 和 Harry 可能全都认为他们各自得到了至少 1/3 的蛋糕并对此感到满意,但 Tom 仍然可能觉得 Dick 的那一份比自己的大。Tom 是一份是"公平"的,但他现在并不觉得很高兴。如果某种分法使任何人都不觉得其他某个人所得到的一份更大,那么这种分法就是"无妒忌的(cnvy-frcc)"。无妒忌分法肯定是公平的,但公平的分法不一定是无妒忌的。因此,找到一种无妒忌分配的算法比找到一种公平分配的算法更困难。

两个人分蛋糕时的"你分我选"的方法显然是无妒忌算法,但上面提到的其他算法全都不是无妒忌算法。60 年代初,John Sclfridgc 和 John H. Conway 首先发现了三个人分蛋糕时的一种无妒忌算法:

第一步,Tom 把蛋糕分成他认为具有等值的 3 块。

第二步,Dick 可以采取下列两个行动之一:

(a)如果他认为其中两块或两块以上的蛋糕同为最大的一份,他可以什么也不做;

(b)如果他认为有一块蛋糕最大,他可以对其进行修整,使之达到(a)所说的情况。把修整下来的零碎蛋糕放到一边。并称这些累积起来的零碎蛋糕为"剩余物"。

第三步,Harry,Dick 和 Tom 依次选一块蛋糕,也就是他们认为最大的一块或并列最大的一块。如果 Dick 在上面第二步中修整了一块蛋糕,那么他必须选修理过的蛋糕,除非 Harry 已经先选这一块蛋糕。

到这一步时,蛋糕的一部分已经用一种无妒忌方式分完了。这样剩下的事情就是通过无妒忌方式把剩余蛋糕也分完。

第四步,如果 Dick 在第二步什么也没有做,那么不再存在剩余



蛋糕,因此蛋糕已被分完。否则,被修理过的那块蛋糕将由 Dick 或 Harry 选去。假定 Dick 得了这块蛋糕(如果是 Harry 得了这块蛋糕,那么从现在起在下面的分法中就把两个人的名字交换)。然后由 Dick 把剩余蛋糕分成他们认为是相等的 3 份。

第五步,剩下的事情就是让 Harry,Tom 和 Dick 依次从剩余蛋糕中各取一份。由于 Harry 是第一个选,因此他没有理由眼红。无论剩余蛋糕怎么分,Tom 都不会妒忌 Harry,因为 Harry 充其量也只能选到一份 Tom 认为是等于 1/3 块的蛋糕。Tom 也不会对 Dick 眼红,因为他比 Dick 先选。Dick 也没有理由抱怨,因为剩余蛋糕本来就是他自己分的。

在这一点上,人人都被难住了 30 年。对于 n 个人分蛋糕的情形,是否存在一种无妒忌分法泥? 1995 年,纽约大学的 Steven J. Brams 以及联盟学院的 Alan D. Taylor 发现了一种引人注目的适用于任意多个人的无妒忌分法。这一方法很复杂,因此我不能在此介绍给读者们。读者可以参看他们的论文"一种无妒忌分蛋糕方法"(见《美国科学月刊》(American Mathmatical Monthly),102 卷,1995 年 1 月),或参看 Jack Robertson 与 William Webb 的精彩著作"分蛋糕算法"(马萨诸塞州拉蒂克市 A. K. Peters 出版社 1998 年出版)。

分蛋糕理论的最令人感兴趣的特点之一就是 Robertson 和 Webb所谓的"分歧有利"论(serendipity of disagreement)。乍看起来,当每个人对每一小块蛋糕值多少的看法都一致时,公平分配似乎是最容易的。毕竟,在这种情况下,大家对于某一份蛋糕的价值不可能有争议了。但实际上情况正好相反:一旦参与分蛋糕的人对蛋糕价值的看法出现分歧,那就更容易使所有人皆大欢喜。

例如,假定 Tom 和 Dick 采用"你分我选"的方法来分一块蛋糕。Tom 把蛋糕分成两份,并认为这两份蛋糕有相等的价值,即每份均为整块蛋糕的一半。如果 Dick 也同意 Tom 的估计,那就不能再做



什么了。但是,假定 Dick 认为这两种蛋糕的价值各为 3/5 和 2/5。这样,出于某种利他主义的理由,他可以决定把他认为是较大的一份蛋糕的 1/12 奉送给 Tom(他估计这一小块蛋糕的价值为整块蛋糕的 1/20)。根据他自己的估价,他仍然还有整块蛋糕的 3/5 - 1/20 = 11/20。实现此分法的一种办法是,Dick 把他认为是较大的一份蛋糕分成他认为具有相等价值的 12 份。然后他请 Tom 从这 12 份中任选一份。

无论 Tom 选了哪一份,Dick 仍然认为他得到了整块蛋糕的 11/20。另外,Tom 有 12 小块蛋糕可以选择,而且他认为这些蛋糕合起来就相当于整块蛋糕的一半。因此,按他的估计,其中至少有一份相当于整块蛋糕的 1/24。他选择了这份蛋糕后,就自以为他得到的蛋糕总共至少是整块蛋糕的 13/24。这样 Tom 和 Dick 现在皆大欢喜:他们都认为自己得到了比公平的一份还要多的蛋糕。

这个问题上的直觉并不是说关于价值的分歧必定会导致关于何为公平分配的分歧。如果由第三方来分蛋糕,然后坚持要 Tom 和 Dick 接受这些已预先确定的某一份蛋糕,那就可能出现这种情况。

遗憾的是,并非所有任务都能够公平地分配,至少是在受到一些合情合理的限制的情况下不能公平分配。以洗碗为例,如果每个人都必须洗净或者烘干一个完整的碗碟,那么在极端的情况下是不可能作到公平分配的。设想有两个人来洗两个碗,其中一个是大碗,而另一个是小碗。两个人显然都愿意洗小碗而不肯洗大碗。因此,即使在一个所有争议都将通过协商来解决的某些分歧来看仍是不可避免的。

17. 人择 Murphy 原理

我从未见过一片烤面包，尤其是又长又宽的烤面包，它在落到满是沙子的地板上时，不是涂奶油的一面先着地。

这是诗人 James Payn 在模仿 Thomas Moore 时写下的几句诗。Payn 所描写的情景是 Murphy 法则的一个典型例子，这一法则宣称，如果某件事情有可能出错，则它就将会出错。美国空军的一位上尉——猜中他的名字不会得到什么奖励——在 19 世纪 40 年代后期就观察到了这种现象。他的原则有多种不同说法和推广形式，例如“即使这不大可能出错，它仍然会出错”等等。除了 Murphy 法则外，这一预言还有其他多种名称。1991 年，英国广播公司(BBC)的电视连续系列节目 QED 差一点推翻了此法则。

该节目的主持人在各种不同条件下把烤面包抛向空中 300 次，并发现烤面包并不特别倾向于涂奶油的一面朝下着地。事实上，其结果从统计上看与随机性是无法区分的。如果不是 Robert Mattews 的话，事情可能就到此为止了。Robert Mattews 是一位有数学爱好的英国记者。他最近在《欧洲物理学杂志》上发表文章，阐述了 QED



实验所存在的两个问题。第一个问题是,Murphy 法则从其本质上说可能促使人们篡改任何旨在对它进行检验的实验。第二个问题是,烤面包在吃早饭时一般不是扔向空中后落到地面上;在大多数情况下它横向地被碰出桌子边缘而掉下去。QED 的实验本应模仿这样一种运动状况。

在继续谈下去之前,必须指出,在通常的一片烤面包中,奶油所占的重量不超过烤面包总重量的 10%。这一添加的物质大部分被吸进了面包的中部。因此,奶油对在空中飞行的烤面包的动力学特性几乎没有什么影响。奶油对烤面包的空气动力学的影响——这一影响源自烤面包表皮上表面粘度的变化——更是可以忽略不计。Mattews 进而把 Murphy 法则追溯到一种较简单的不对称性上:由于烤面包的上部表面涂了奶油,因此当它被碰出桌子的边缘时这同一面始终在上面。

当烤面包向地面落下时,它以某一角速度转动,此角速度取决于当它开始下落时它的质心悬挂于桌子边缘上的角度。情况是不是一般桌子的高度和地球引力的作用共同促成了烤面包通常都转动 180 度的奇数倍呢?根据 Mattews 的计算,简捷的回答的确如此。事实上,最常见的情况是,它的转动刚好使烤面包翻转一次,结果就造成它涂奶油一面向下着地了。

在考察这一不幸巧合的更深层的原因之前,我们最好先总结一下 Mattews 的数学论证。附图示出了烤面包以及有关的主要变量的初始情况,还有从牛顿运动定律推导出的某些关键公式。主要的结论是,烤面包不可能涂奶油一面向上着地,除非"临界突出参数"——即烤面包在掉下之前其伸出桌子边缘的部分与面包块的宽度的一半之比——至少为 6%。实验证明,对于面包,这一数值为 2%,而对于烤面包,这一数值为 1.5%。这两个数值都太小,不足以使面包或烤面包在向下掉到地板的过程中转动至少 360 度。由于可以证明此转



动至少为 180 度,因此涂奶油一面朝下着地是不可避免的结果。

Mattews 的论证作了一些假设,其中一个假设是烤面包一旦触地后不会反弹。毕竟典型的情况是面包"呼"的一下拍在地上而不是"呼"的一下着地。另一假设是烤面色慢慢地滑出桌子边缘,这样它将经临界伸出值脱离桌子。这个假设也不会遇到什么大麻烦。除非烤面包在滑出桌子边缘时其水平速度至少为每秒 1.6 米——等于是猛地一下冲出去——否则这一速度对烤面包的下落不会有什么大的影响。因此,如果你发觉烤面包滑出了饭桌,你可以用手使劲打它一下。这一方法或许无助于救出烤面包,但可使地毯免于沾上奶油。

上述分析是很不错的,但它暗示 Murphy 法则纯属巧合,是人类文化带给饭桌和面包的若干任意值与地球重力场的同样任意的值相结合而产生的所谓"murphy 共振"的一个怪异实例。事实上,正如 Mattews 所证明的那样,没有什么比这离真理更远的了。体现于扔烤面包之中的 Murphy 法则是自然界基本常数的一个深刻的结果。任何存在着与我们极不相似的生物的宇宙也必定把 Murphy 法则强加给它的居民——至少是如果这些生物也吃烤面包并坐在桌旁吃饭的话。

精确的论证是技术性的,相当复杂,但其要点却比较简单。关键的资料来自哈佛大学的 William H. Press。他在 1980 年提出,用两脚行走的生物的高度受它生活在其中的引力场的限制。与用四足行走的动物相比,两足动物在本质上是不稳定的。它们的质心只要稍不小心移到了其"脚印"之外便会使它们摔倒。四足动物的稳定性区域便要大得多。长颈鹿比人高并非巧合。

两足动物的最大高度是如其摔倒则头部所受的伤害可能致死的高度。有理由假定,由这样一种有智慧的两足动物使用的桌子的高度大约为其自身高度的一半。在地球上,一张桌子必需有大约 10 英尺高才会使 Murphy 法则不成立,这样一来我们必须有将近 20 英尺



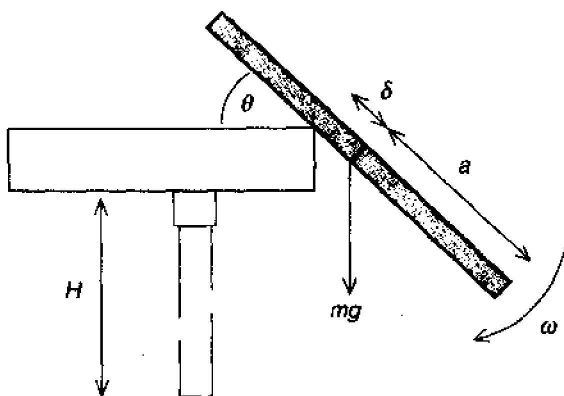
高才能摆脱 Murphy 共振的不幸结局。真正的问题是:某个遥远行星上的外星人是否可能不受 Murphy 法则的支配?

为了找到答案。Matthews 建立了一个外星人模型,把外星人当作一个圆柱形聚合物,其顶端为一个似头的球。我把这种生物称作聚 Murphy。如果一个聚合物层中的化学键断了,这个外星人就会死去。Matthews 确定,一个活着的聚 Murphy 的高度至多为$(3nq/f)^{1/2}\mu^2A^{-1/6}(\alpha/\alpha_G)^{1/4}a_0$。在这个式子中,n 为发生破裂的平面中的原子数目(通常是 100 左右);q 等于$(3.10)^{-3}$,这是一个与聚合物有关的常数;f 是使聚合物键断裂的那部分动能;μ 是聚合物原子的半径(其单位的玻尔半径);A 是聚合物材料的原子量;α 等于电子精细结构常数 $e^2/2h\varepsilon_0c$。其中 e 为电子的电荷,人为普朗克常数,ε_0 是自由空间的电容率,c 为光速;α_G 是引力精细结构常数 $2\pi Gm_p/hc$。其中 G 为引力常数,m_p 是质子的质量:a_0 为玻尔半径。

把我们所在的宇宙的各个有关数值代入上式,便求得一个聚 Murphy 的最大安全高度为 9 英尺 8 英寸(顺便说一下,有案可查的最高的人是一个叫 Robert Wadlow 的人,其身高为 8 英尺 11 英寸)。这远远不到为了不让厨房地板让上奶油,需要的 20 英尺。有意思的是,聚 Murphy 身高的这一上限与它住在哪个星球上无关。其理由在于,保持聚 Murphy 不致分崩离析所需的内部引力作用与静电和电子简并效应之间的平衡把行星的引力同某些更基本的常数联系了起来。因此,我们发现 Murphy 法则完全不是巧合,而是某种深刻的“人择 Murphy”原则:任何按通常方式建造起来的,有智慧聚 Murphy 居住的宇宙都将符合 Murphy 法则。Matthews 的结论是,“爱因斯坦认为,上帝是高深莫测的,但他并无恶意。或许是如此吧。但上帝对下落的烤面包的作用却明显地大有可改进之处。”

烤面包的Murphy动力学

定义临界伸出常数为η(临界伸出常数是烤面包的初始伸出部分除以烤面包宽度的一半)。然后,根据牛顿运动定律,只要烤面包以饭桌边缘为枢轴转动,就可得出关系式$\omega^2=(6g/a)(\eta/1+3\eta^2)\cdot\sin\theta$。当烤面包的重量超过了饭桌边缘的摩擦力时,烤面包就开始下滑。在这一瞬时的转动速度就是烤面包随后在其下落过程中的转动速度。



简单的估计表明,烤面包在掉向地板的过程中将翻转至少180度。因此,要使它着地时涂奶油的一侧朝上,它至少必须转动360度。我们知道烤面包转动速度有多快,而H(连同g)则告诉我们烤面包需多长时间才落到地板上。对于通常大小的饭桌和烤面包,Matthews证明,烤面包仅在临界伸出参数大于0.06时才转过至少360度。临界伸出是烤面包刚脱离桌子并开始自由下落时发生的。

g=重力加速度

m=烤面包的质量

a=烤面包的宽度的一半

δ=初始伸出



θ= 转动角

ω= 转动的角速度

H= 饭桌的高度

18. 黄金数与塑料数

先谈谈黄金数 $\varphi=1+1/\varphi=1.618\ 034$(近似值)。黄金数与著名的斐波那契数有密切的关系。斐波那契数列可以用排成螺旋状的一系列正方形来说明(见图 1 的上图)。起始的正方形(图中用灰色表示)的边长为 1,在其左边的那个正方形的边长也是 1。在这两个正方形的上方再放一个正方形,其边长为 2,以后顺次加上边长为 3、5、8、13、21 等等的一系列正方形。这些数字的每一个均等于前面两个数之和,它们构成了斐波那契数列。相邻的两个斐波那契数的比值逐渐趋近于黄金数,例如 $21/13=1.615\ 384$

这一事实可从生成斐波那契数的规则推导出来:对于较大的数,由此规则可得出方程 $\varphi=1+1/\varphi$。如果在每个正方形内画一段相当于圆周的 1/4 的弧,那么这些弧合起来就形成一条优美的螺旋线。这条螺旋线与自然界中经常发现的所谓对数螺线很相似(例如鹦鹉螺的壳就是对数螺线的形状)。它的各圈的长度以约等于黄金数的比率增加。

上面讲的是黄金的故事,下面要讲的就是塑料的故事了。我们

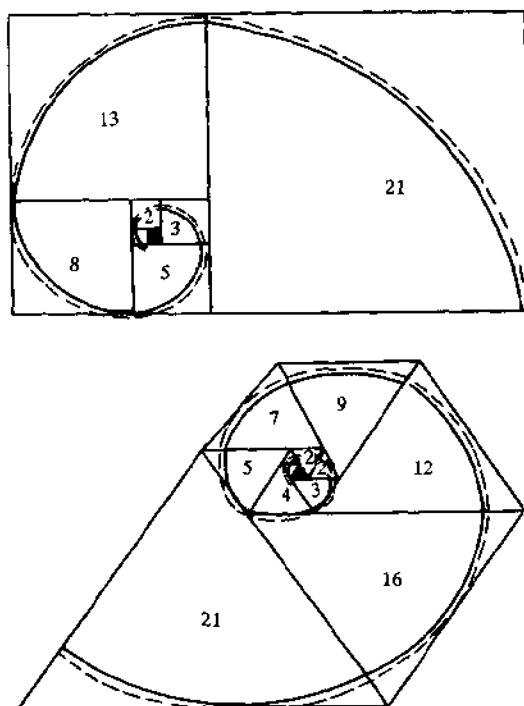


图 1　用螺旋线说明斐波那契数(上)和 Padovan 数(下)的生成过程

从一幅类似的图着手,但它是由一些等边三角形构成的(见图 1 的下图)。初始三角形用灰色表示。后面的三角形沿顺时针方向依次加上去。产生的螺旋线又差不多是对数的。为了使这些三角形天衣无缝地拼在一起,头三个三角形的边长均为 1。其后的两个三角形的边长为 2,然后依次是 3、4、5、7、9、12、16、21 等等。这个数列也有一条很简单的构成规则:每个数都是跳过其前面的那个数,并把再前面的两个数相加而得出的。我把这个数列称为 Padovan 数列,以纪念建筑师 Richard Padovan(奇妙的是,Padova 是 Padua(帕多瓦)的意大利文形式,而斐波那契则来自比萨——距帕多瓦约 100 英里)。



用代数符号，可以把斐波那契数列 $F(n)$ 和 Padovan 数列 $P(n)$ 分别表示为：$F(n+1)=F(n)+F(n-1)$，其中 $F(0)=F(1)=1$，和 $P(n+1)=P(n-1)+P(n-2)$，其中 $P(0)=P(1)=P(2)=1$。这两个数列之间的相似性是非常明显的。塑料数——从现在起我将把这个数称为 P，其近似值为 1.324 718——是相邻的两个 Padovan 数之比的极限，正如黄金数是相邻的两个斐波那契数之比的极限一样。由构成规则可得方程 $P=1/P+1/P^2$，即 $P^3-P-1=0$。数 P 是这个方程唯一的实数解。

由于 P 小于 φ，因此 Padovan 数列的增长速度比斐波那契数列慢得多。Padovan 数列存在许多有趣的规律。例如，由图 1 中可以看出 $21=16+5$，因为在同一条边上的相邻三角形必须能拼合在一起。这样，可以得出推导 Padovan 数列更多项的另一条规则：$P(n+1)=P(n)+P(n-4)$。

某些数既是斐波那契数，又是 Padovan 数，例如 3，5 和 21。还有其他的例子吗？如果有的话，那么这种数有多少，而且是有限还是无限呢？某些 Padovan 数是完全平方数，如 9，16 和 49 等，还有其他 Padovan 数也属于这种情况吗？这三个数的平方根（即 3，4 和 7）也是 Padovan 数。这是巧合呢还是一种普遍规律？这些问题以及其他许多问题都值得进一步研究。

生成 Padovan 数的另一种方法是模仿用正方形来生成斐波那契数的方法，但用长方体（即各面为矩形的盒子）来代替正方形。这样我们就得到由长方体形成的某种三维螺旋线（见图 2）。开始时是一个棱长为 1 的立方体，在其旁边也放一个棱长为 1 的立方体，其结果就是一个 $1\times1\times2$ 的长方体。在这个长方体的 1×2 的一面上，加上另一个 $1\times1\times2$，得到一个 $1\times2\times2$ 的长方体。然后在它的 2×2 的一面上，加上一个 $2\times2\times2$ 的立方体，从而总共形成一个 $2\times2\times3$ 的长方体。在这个长方体的 2×3 的一面上，加上一个 $2\times2\times3$ 的长方



体,得到一个 2×3×4 的长方体,如此类推。把这一过程继续进行下去,始终按照东→南→下→西→北→上的顺序加上新的长方体。在每一步上,新形成的长方体均以三个连续的 Padovan 数为其三条棱的长度。

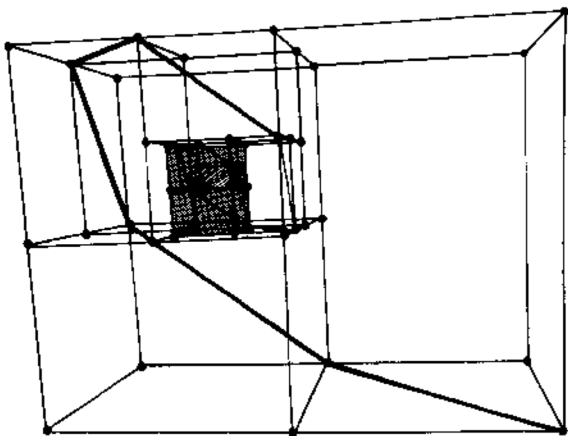


图 2　成螺旋状排列的长方体也可构成 Padovan 数

此外,如果你把新加长方体的正方形面依次用直线连接起来,那么就得到一条螺旋线。甚至还可以证明这条螺旋线在一个平面上。

法国数学家 Edouard Lucas 在 1876 年研究了具有相同的构成规则,但其初始数值不同的一个数列。1899 年,R. Perrin 进一步发展了他的想法,现在这个数列被称为 Perrin 数列 A(n)。Perrin 数与 Padovan 数不同的地方在于 A(0)=3,A(1)=0,A(2)=2。相邻的两个 Perrin 数之比也是趋近于 P,但是 Lucas 注意到一个更为微妙的性质。当 n 是素数时,它便可整除 A(n)。例如,19 是一个素数,A(19)=209,而 209/19=11。

这个定理提供了检验某数是不是合数——即非素数——的一种奇妙方法。例如,当 n=18 时,我们有 A(18)=158,而 158/18=



8.777,其结果不是整数。因此,18 必定是合数。这样我们可以用 Perrin 数来作非素性检验:任何不能整除 A(n)的 n 必定是一个合数。

如果 n 整除 A(n),那么 n 必定是素数吗? 从 Lucas 定理并不能得出这个结论,正如从"如果天下雨,我就会被打湿"不能推出"如果我身上打湿了,那么天就在下雨"一样(我可能是在风和日丽的一天里掉入池塘中而打湿了的)。不过,这是一个很吸引人的悬而未决问题。没有任何人发现了一个可整除 A(n)的合数 n,但也没有任何人证明了这种数——称为 Perrin 伪素数——不存在。1991 年,马里兰州鲍威超级计算研究中心的 Steven Arno 证明了 Perrin 伪素数必定至少有 15 位。

不存在 Perrin 伪素数的猜想具有重要意义,因为用 n 除 A(n)所得的余数可以很快地计算出来。如果这一猜想正确,那么,当且仅当 n 为素数时,余数为 0,这样就提供了一种快捷的素性检验法(事实上,1982 年,马里兰大学的 William W. Adams 和 Daniel Shanks 发现了通过 logn 步骤快速计算此余数的一种方法)。因此,这一猜想对于密码应当有很大用处,因为现在的密码常常与大素数的性质有关。

正如闪闪发光的黄金数一样,平淡无奇的塑料数也产生了丰富多彩的螺旋设想。

19. 算术与鞋带

至少有3种一般的系鞋带方式(见图1),即美式之字形系法,欧式直系法(术语"Straitlaced"(古板的)即源出于此,不过可能是指的服装而不是鞋子),以及鞋店式快系法。对于买鞋子的人来说,各种系鞋带方法的美观程度可能不同,系鞋带所需的时间也不同。对于制鞋厂来说,它们更关心的问题是何种系鞋带方式所需的鞋带最短,因而花费也最少。本文将考虑制鞋厂的问题。

为了求出所需鞋带的量,我们只考虑直的线段所代表的长度。把鞋带系成一个蝴蝶结所需的额外鞋带的长度对所有各种系法来说都是一样的,因此可以忽略不计。

本文使用的术语都指的是从穿鞋者的角度来看的系鞋带,因此,图中的"顶上"一行鞋眼靠近脚踝。同时我们将把鞋带理想化为数学上的线,其厚度为零,而鞋眼则理想化为点。采用硬算的办法,可以根据以下三个参数计算出鞋带的长度:

- 鞋眼的对数 n
- 相邻鞋眼间的距离 d

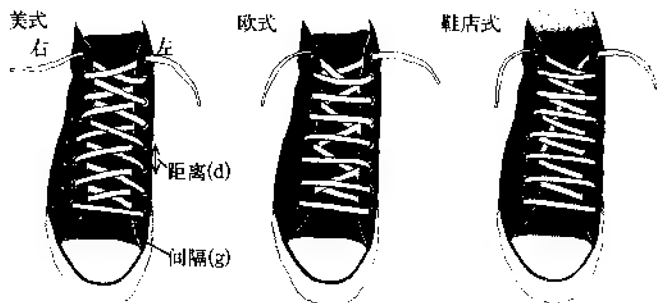


图 1　旅游鞋上的鞋带系法其复杂性和长度各不相同。哪种系法所需要的鞋带最少

- 对应的左鞋眼与右鞋眼之间的距离 g

借助毕达哥拉斯定理,不难证明鞋带的长度可用下列公式计算:

- 美式:$g+2(n-1)\sqrt{d^2+g^2}$
- 欧式:$(n-1)g+2\sqrt{d^2+g^2}+(n-2)\sqrt{4d^2+g^2}$
- 鞋店式:$(n-1)g+(n-1)\times\sqrt{d^2+g^2}+\sqrt{(n-1)^2d^2+g^2}$

哪种系法的鞋带最短呢?为便于论证,设 $n=8$(如图 1 所示),$d=1,g=2$。据此求出鞋带的长度:

- 美式:$2+14\sqrt{5}=33.305$
- 欧式:$14+2\sqrt{5}+6\sqrt{8}=35.443$
- 鞋店式:$14+7\sqrt{5}+\sqrt{53}=36.933$

但是我们能肯定美式系法总是最短的吗?仔细地用高中代数进行一番计算,可以证明如果 d 和 g 是非零数,且 n 至少为 4,则最短的鞋带总是美式系法的鞋带,其次是欧式鞋带,再次是鞋店式鞋带。如果 $n=3$,则美式鞋带仍是最短的,但欧式与鞋店式鞋带的长度则相同。如果 $n=2$,则所有三种系法的鞋带长度都相同,不过只有数学家才会为这些情况操心。



然而，这一代数方法比较复杂，而且不能说明是什么原因使不同的系法具有不同的效率。运用一种巧妙的几何技巧，就能一眼看出为什么美式系法的鞋带是3种系法中最短的。这一想法得自于光学——研究光线路径的学问——的启发。

数学家们早就发现，运用仔细选择的反射来把弯曲的光线路径拉直，就可以使光线的几何性质变得可以说是更透明。例如，为了导出经典的反射定律——"入射角等于反射角"——试考虑一束射到镜子上然后从其上反射的光线。如果你作出光线后半段路径关于镜子所在平面对称的反射像（见图2），则结果就得到一条穿过镜子并进入镜子后面Alice所在世界的路径。

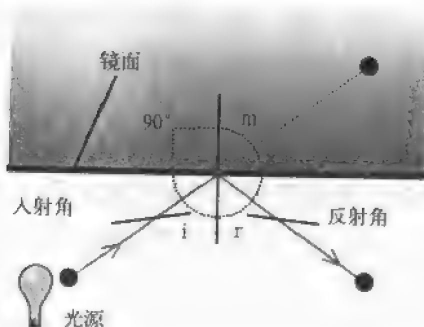


图2　导出光线路径的费马原理表明，反射角等于入射角

根据最少时间原理——这是费马所阐明的光线的一条普遍性质——这样一条路径必须在最短时间内到达其目的地。在本例中，这就意味着它是一条直线。在光的入射点作一条与镜面垂直的线。这样图中标有m的角就与入射角i相等。而且，标有x的两个角相等。但$m+x=90°$；如果r是反射角，则$r+x$也为90°。因此$m=r=i$。

借助于上述反射技巧的一种推广形式，可以推导出所有3种鞋带系法的几何表示。在一张图上画出2n列鞋眼，鞋眼在垂直方向上



的间距为 d，在水平方向上的间距为 g（见彩图 3）。为了把图缩小，我们减少了 g 的值；无论 d 与 g 取何值，此方法都行得通。图中最后一列代表鞋子的左边一列鞋眼，倒数第二列代表右边一列鞋眼。一般地说，编号为奇数的列代表左边一列鞋眼，而编号为偶数的列则代表右边一列鞋眼。

在这幅图上蜿蜒穿行的折线路径对应于各种鞋带系法，但有一点额外的“花样”。从某一系法的左上角鞋眼出发，从鞋子的左边到右边在本图的第 1 列和第 2 列之间画出第一段鞋带。接下来在第 2 列和第 3 列之间画出下一段鞋带，而不是像真正的鞋子那样从第 2 列返回到第 1 列。事实上，这一段鞋带被反射到另一侧，似乎各列鞋眼被镜子取代了一样。如此继续进行下去，每遇到一个鞋眼时就依次把一段鞋带反射到另一侧。这样，鞋带的路径就不是在两列鞋眼之间往返穿行，而是一直延伸到图的左侧。

由于一段鞋带的反射并不改变它的长度，因此这一表示法所得的路径的长度与相应的鞋带系法的鞋带长度是完全相等的。然而此表示法还有一个好处，就是很容易比较出美式系法和欧式系法的鞋带长度。有几个地方这两种系法的鞋带彼此重合，但在其他所有地方，美式系法的鞋带穿过一个小三角形的一条边，而欧式系法的鞋带则穿过该三角形的另外两条边。由于三角形的任意两边之和大于第三边的长度（也就是说任意两点间的路径以连接两点的直线为最短），因此很明显美式系法的鞋带要短一些。

鞋店式系法的鞋带比欧式系法的鞋带长则不是很明显的。看出这一点的最简单方法是从两条路径中去掉所有水平线段（每条路径有 n－1 条水平线段，其长度相同）以及长度相等的两条倾斜线段。去掉这些线段后，就得到两条 V 字形路径。把去掉了第一条线段的欧式系法的路径平移，使其又同鞋店式系法的路径一样都从第一个鞋眼出发。如果把每条 V 字形路径都以其顶点处的一条水平轴为



中心反射到另一侧从而使它们拉直，最终就容易看出鞋店式系法的路径要长一些，这也是因为三角形两边之和大于第3边。

这些聪明的反射技巧并不只是可以比较各种系法的鞋带长度。应用这些技巧，可以证明在所有可能的系法中，美式之字形系法的长度是最短的。更一般地，鞋带和费马式光学在测地线——即各种几何中的最短路径——的数学理论中统一了起来。在这一理论中，镜面反射除了证明美式鞋带系法的优越性外，还解决了物理学中的基本问题（见彩图4）。

20. 半真半假的故事

让我作一下自我介绍。我名叫 Epimenides，是一个职业的说谎者。嗯，那倒不完全是真的。我的名字其实是叫 Herman Fenderbender，在一家汽车保险公司工作。但是我在“悖论俱乐部”(Paradox Club)的朋友们称我为 Epimenides；当我同他们在一起时，我总是说谎。

上个星期四天在下雨，因此我到俱乐部稍微晚了一点。苏格拉底和柏拉图靠在柜台上，他们的旁边是一个圆脸蛋的小家伙。

柏拉图说：“这是我们俱乐部的新成员 Lukasiewicz。”

我不高兴地说：“见到你真是糟糕。我叫齐诺。”

苏格拉底解释道：“他的意思是说他很高兴见到你，他的名字叫 Epimenides。他总是说谎。”

“那不是真的”，我说。然后我打开皮夹子，掏出我的名片。“这不是我的名片”，我宣称道，然后将名片递了过去。Lukasiewicz 看见名片上的一面写的是：“本名片另一面上的话是真的。”他把名片翻过来，又看到这样一句：“本名片另一面上的话是假的。”



我得意洋洋地说:"不过苏格拉底是对的。我老是说谎。"

Lukasiewicz 热情地同我握手。"我很高兴见到你,不过这话有1/3是假的。你的名片上两面的话都有一半是真的。"

"什么?"我说。

柏拉图解释道:"Lukasiewicz 对模糊逻辑很感兴趣。"

Lukasiewicz 说:"现在的规定是真语句的真值为1,假语句的真值为0,我打算不限于只考虑这两种情况,还要考虑真值为0.5的'半真'情况和真值为0.1的'几乎假'情况,一般说来,就是真值可以为0到1之间的任何数。"

我茫然不解地问:"为什么会有人想要考虑这个?"

Lukasiewicz 笑了起来。"假如我说俱乐部主任很像查理·卓别林,你看我这话是不是真的?"

"当然不是!"

"连他的脚也不像吗?"

"嗯,我想脚有点儿……"

"这么说来我的说法也不完全是假的。"

"嗯,他的确有点像卓别林。"

Lukasiewicz 俯身靠近我。他的目光灼灼逼人。"倒底有多像卓别林?"

"我看大约是15%。"

"很好。这么说我的陈述'俱乐部主任看起来像查理·卓别林'有15%是真的。在模糊逻辑中,它的真值为0.15。"

"这只不过是玩弄文字游戏罢了,它没有什么意义的。"

Lukasiewicz 一把抓住我的手臂。"不,它有意义。它有助于解决悖论。例如,你宣称自己是个彻头彻尾的说谎者。让我们考虑一下你的陈述'我在撒谎',或者更简单些'这个陈述是假的'。在经典逻辑中,这是一个悖论,对吧?如果此陈述为真,那么它就是假的;如



果此陈述为假,那么它就是真的。换言之,你有一个其真值 p 为 0 或 1 的语句 P,而 P 宣称此语句的真值为 1 - p。"

"抱歉,我听不懂你说些什么。"

"喔。如果 P 为真,则其反面(即非 P)就为假,其真值即为 0。反之亦然。而 1 - 0 = 1 且 1 - 1 = 0,因此,如果 P 的真值为 p,则非 P 的真值为 1 - p。"

"喔,我明白了。"

"对。现在的问题是'本语句'是 P,因此 P 告诉我们 P 的真值是 1 - p。这就导致了悖论。如果 p = 0,那么 P 告诉我们 p = 1 - 0 = 1,而如果 p = 1,那么 P 告诉我们 p = 1 - 1 = 0。这两种选择都不是无矛盾的。"

我对他报之以轻蔑的一笑。"Luke,你所做的一切不过是用复杂的代数语言把本来一直十分明了的问题重新描述一番而已。"

他勉强地笑了起来。"或许如此吧。不过,在模糊逻辑中,方程 p = 1 - p 有一个无矛盾的解,即 p = 0.5。因此你自称是一个无时不说谎的人的说法为半真,这样一切问题都迎刃而解了。你自己的说法不可避免地导致了模糊逻辑。"柏拉图拍了拍他的背,而苏格拉底则笑得几乎碰翻了鸡尾酒。我的脸一下子变红了,不过我看到了问题的核心。

"他的名片又如何呢?"柏拉图问。Lukasiewicz 正要说话时被我挡住了。"让我来回答这个问题。在我看来,我有两个陈述 P 与 Q,其真值分别为 p 与 q。此外,P 宣称 Q 是真的,而 Q 宣称 P 是假的。因此相应的真值方程是:

对 P 为 $p = q$

对 Q 为 $q = 1 - p$

如果 p 与 q 的取值仅能为 1 或 0,则此方程组无解。但在模糊逻辑中此方程组有一个唯一的解:$p = q = 0.5$。因此我的名片的每一面上的



话都是半真的,这样就再也不存在悖论了。"

"非常正确",Lukasiewicz 说,"但是还不止于此。迄今我们所讨论的仅是动态逻辑这一整套新理论的开端则已。这种逻辑是纽约州立大学石溪分校哲学系的 Gary Mar 和 Patrick Grim 发明的,它把语义悖论和混沌理论联系起来了。"

这下轮到苏格拉底茫然不解了。

"欧,清醒一点吧。你知道混沌是什么。混沌就是简单的确定性的动态规则导致无规则的看起来随机的行为,蝴蝶效应,以及诸如此类的东西。"

"这个我当然知道,"苏格拉底恼火地说道,"把我搞糊涂的是动态逻辑这一观念。逻辑怎么会是动态的呢?"

Lukasiewicz 看起来似乎感到吃惊。"在讨论自指示语句时,逻辑又怎么会不是动态的呢? 语句自身迫使你修改对语句真值的估计。这一被修改了的值随后又得一而再、再而三地修改。试考虑'说谎者悖论',也就是你的语句 P:'这一语句是假的。'先前我已写出了这一语句的真值方程 $p=1-p$。但是实际上我写下来的本应该是一个迫使你不断修改你对真值的估计的过程,即 $p\leftarrow 1-p$。如果你假定 P 有某一真值 p,则 P 自身就告诉你用 $1-p$ 替换该真值。例如,如果开始时你认为 P 有 30% 是真的,则 $p=0.3$,由修改规则得出 $p=0.7$,然后又得出 $p=0.3$……这样你就得到一个真值的无穷序列,此序列在 0.3 和 0.7 这两个值之间作周期性振荡。经曲悖论中 p 只能为 0 或 1,这样就得出 0,1,0,1……这样一个序列,它如实地反映了下面这个逻辑论证过程:如果 P 为假,则 P 为真,则 P 又为假,则 P 又为真……真值的动态特性体现了悖论的逻辑振荡。"

"那么 $p=0.5$ 就是唯一的一个不会引起振荡的值了",柏拉图沉思着说。

"不错。现在我们来看你的名片上的二元悖论吧。这个悖论的

确是一个逻辑动态方程:

$p \leftarrow q$

$q \leftarrow 1 - p$

假定开始时你估计 $p=0.3, q=0.8$。然后你对这两个值作第一次修改,得 $p=0.8, q=0.7$。再作一次修改后得出:$p=0.7, q=0.2$。第三次修改后得出 $p=0.2, q=0.3$,第四次修改得 $p=0.3, q=0.8$,这样你又回到开始的初值上了。这个方程的解以 4 为周期反复循环,除非你取 $p=0.5$ 和 $q=0.5$ 为初值。在这种情况下,p 和 q 的值就永远也不变。"

"好,我相信这一点",我说,"但是混沌又是怎么回事呢?"

Lukasiewicz 的表情变得非常严肃。"在开始解释混沌之前,我得把叙述弄得更准确一点",他说,"如果你想自己玩一下这些概念,我最好是告诉你如何计算逻辑语句组合的模糊真值(见本文后面框内的文字)。不过眼下你真正需要知道的只是:如果 P 的真值为 p,则非 P 的真值为 $1-p$。其次,你必须知道如何估计关于语句的语句的真值。"

"请举个例子。"苏格拉底说。

"可以。假定我说柏拉图是个出色的高尔夫球手。你看这话有几分真实性?"

"哦——,大概 40% 吧",苏格拉底说。柏拉图恶狠狠地盯了他一眼。"嗯,Epimenides 常常赢你,而他不过是个相当平庸的球手而已。"我用更凶狠的目光盯了苏格拉底一下。

"很好。且让我们称此语句为 S。它的真值是 $s=0.4$。假定我作出一个有关语句 S 的语句,即语句 T:'S 是 100% 真的。'那么语句 T 的真实性有多大?"

我思索了片刻。"嗯,它本身肯定不是 100% 真的。否则 S 就是 100% 真的了,而我们已经判定 S 不是 100% 真的。"



"不错。我的这个关于语句S的语句T的真实程度取决于S的实际真值和T赋与S的真值。这里$s=0.4$,但是T诱使我去评估出的值却是1。因此,只要这两个值不同,T就是非真的,对吧?我的估计越不精确,我的语句就越假。由于这两个值现在相差0.6,因此T的假的程度就为0.6。也就是说它的真的程度为0.4。"

"如果你说'S是半真的',那么情况又如何呢?"

Lukasiewicz高兴地点点头。"你将会看到这个问题将得到很好的解决。你刚才说的这个语句估计S的真值为0.5,但实际真值是0.4。这两个值的差为0.1,这就是你的语句的假的程度,因此它的真值为0.9。由于你的估计仅错了10%,因此你对的程度为90%。"

"呵。如果你说'S是40%真的',那么我就100%的正确了。这样真值就将为1。我明白了。"

"很好。一般说来,假定我有一个真值为p的语句P和一个引导你去估计P的真值为q的语句Q。那么,根据我们上述的一番论证,可知Q的真值为$q=1-|p-p'|$,这里$|x|$代表x的绝对值(如果x为正,则$|x|=x$,如果x为负,则$|x|=-x$)我把它称为估值公式。"

Lukasiewicz想了片刻。"现在我可以让你见识见识我称之为'混沌说谎者'的语句C:

本语句真的程度同它被估计为假的程度一样大。

如果这一语句的真值为C,则它就引导你去得出真值为$1-c$这一估计值。因此,根据估值公式,它的真值为$1-|c-(1-c)|=1-|1-2c|$。简言之,存在下面这个重新估计真值c的动态过程。

$c\leftarrow 1-|1-2c|$

选择一个数作c的初始值,比如说令$c=0.12345$,并按上式逐次计算出c的各个值。你将会发现c的这一系列值是混沌的。事实上,我得提醒你,由于你用的计算器或计算机中的舍入误差,这一迭代过程可能最终将稳定在0或1上。用下面这个式子来代替上面的

动态方程或许会更好一些：

$$c \leftarrow 1 - |0.999\,999 - 2c|$$

你甚至能够观察到混沌理论的有名的蝴蝶效应——一只蝴蝶拍一下翅膀，就可能在1个月后引起一场飓风。更直截了当地说，初始条件的微小变化将使后来的动态过程发生重大变化。如果你用的初始值是0.123 46而不是0.123 45，那么将会得到一幅不同的图景。"

Lukasiewicz停顿了一下。"接下来还有称为'混沌二元论'的两个语句：

X：X为真的程度与Y为真的程度一样大

Y：Y为真的程度与X为假的程度一样大

这看起来有点像你的名片了，Epimenides。动态公式是：

$$x \leftarrow 1 - |x - y|$$
$$y \leftarrow 1 - |y - (1 - x)|$$

为了看看它能得出什么结果，你可以选择一对初始值，比如说$(x,y)=(0.2,0.9)$，并按上式计算出逐次的(x,y)的值。试把x和y的值看成是坐标，并把它们在平面上绘出来。这样你就得到一种被称为混沌系统吸引子的几何图形，在本选定值下，你得出的是一个密布着小点的三角形(见彩图5右图)。这一图形可转换为一种漂亮而复杂的图像，称为'逃逸时间图'。为了作出这样一幅图，我们暂且把x和y必须位于0和1之间这一条件放宽一下。我们的设想是要观察(x,y)移动到离原点多远的地方，并统计要经过多少步计算后(x,y)才会越过某一域值。然后将点(x,y)用颜色绘出，具体的颜色与所需要的计算步骤的多少有关。开始时应当选择一个刚好大于1的数作域值(见彩图5的左图)。"

"现在我开始明白了"，苏格拉底说道，"你抓住在估计一组自指示语句的真值时所用的一系列思考过程，并把它转化为一个动态过程。然后你就可以把混沌理论的全部方法用于这个过程。逃逸时间



图正是在创造出所有那些与 Mandelbrot 集有关的精巧彩色图像(包括涡旋螺线、海马、仙人掌、星星等)的方法的启示下作出的。"

"的确如此。这里还有最后一个问题可以让你们仔细考虑。我们可以把'混沌说谎者'语句改写为:

本语句的估计的假的程度与其估计的真的程度并无不同。

在模糊逻辑中,把形容词'非常'解释为构成一个真值的平方已成为一项标准的规定。由此再想想下面这个更模糊的语句:

本语句的估计的假的程度与其估计的真的程度相差不是非常大。

这个语句导出下面的动态公式:

$p \leftarrow 1-(p-(1-P))^2$

它可以转化为:

$p \leftarrow 4p(1-p)$

混沌理论家们把它称为逻辑斯谛动态系统,因此我的语句也就是'逻辑斯谛说谎者'。它是混沌的,你可以试试看。"

午夜时分,悖论俱乐部关门了。我和 Lukasiewicz 走出俱乐部来到街上。我意识到,由于我全神贯注于研究模糊逻辑混沌的实例,因此忘记了问一个非常重要的问题。"Luke,所有这些都挺不错的,但是它有多大的意义呢?"

"嗯",他说,"Mar 和 Grim 指出,它提供了一种几何上的研究语义复杂性的方法,使你能够区别出不同的自指示语句系统。他们还指出,它可用于证明不存在一个使你可以知道某一系统是否混沌系统的判定过程。这个结果与 K. 哥德尔的著名的公理不可判定性定理属于同一类型。不过,Epimenides,这些是比较深奥的内容了。"

"这么说来我明白了。把逻辑和混沌联系起来——简直是妙不可言! 不过等一下。我怎么能够确信你告诉我的一切全是真的呢?"

"如果我对你说了假话,我就请求上天用两道闪电把我打倒在



地。”

刚好在这时，雷雨云在天空聚集起来，一道闪电把 Lukasiewicz 打得失去了记忆。我抬头望天，朝乌云挥舞着拳头：“他刚才对我说的话是全部为真，还是只有半真？”

模糊逻辑

在经典逻辑中，一个语句若为真，其真值即为 1，若为假，其真值即为 0。“太阳在照耀”这一语句当天空布满乌云时其真值即为 0。一般地说，语句 P 的真值 p 不为 1 即为 0。在模糊逻辑中，一个语句的真值可在 0 到 1 之间。如果乌云遮住了太阳的 3/4，则语句 P 的真值即为 0.25。

在模糊逻辑中如同经典理论中一样，当施行“非”(NOT)、“与”(AND)、或(OR)，“蕴含”(IMPLIES)和“当且仅当”(IF ANDONLY IF)等操作时，语句的真值将改变。

“非 P”的真值为 $1-p$。

例，如果“太阳在照耀”这一语句的真值为 0.25，则“太阳不在照耀”这一语句的真值即为 0.75。

“P 与 Q”的真值等于 p 与 q 中的较小者(q 为语句 Q 的真值)。

例，“太阳在照耀”(真值为 0.25)与“Jane 正被晒黑”(真值为 0.10)。

这个例子的真值为 0.10。

“P 或 Q”的真值等于 p 与 q 中的较大者。

例，“太阳在照耀”(真值为 0.25)或“Jane 正被晒黑”(真值为 0.10)。

这个例子的真值为 0.25。

“P 蕴含 Q”的真值等于 1 和 $1-p+q$ 中的较小者。

例，“如果太阳在照耀(真值为 0.25)，则 Jane 正被晒黑(真值为 0.10)。”



这个例子的真值为 0.85。

"当且仅当 Q 时有 P"的真值为 $1-|p-q|$，也就是 1 减去 p 减 q 的绝对值。

例，"太阳在照耀(真值为 0.25)，当且仅当 Jane 正被晒黑(真值为 0.10)。"

这个例子的真值为 0.85。

21. 怪数的新命理学

地狱之火的火苗在高耸的烟柱间摇曳着，似乎要把铅灰色的天空烤焦。远处，冥神的铁墙被烧得通红。一个穿着运动衫的绿色妖怪在沸腾的油湖边懒洋洋地闲荡着，百无聊赖地把一个褐色的椭圆形玩意在爪间抛来抛去。运动衫正面印着“666”这个数字，而当妖怪转过身去练习投掷时，可以看见运动衫背面上是“THE BEAST”（怪兽）这个名字。

另一个看起来比较悦目的淡蓝色妖怪拍了拍他的肩膀。“嗯——对不起，Beast 先生。”

“你是谁？”Beast 转过身来，盯着蓝妖怪的运动衫，看见上面印着“$-847\frac{1}{2}$”。“$-847\frac{1}{2}$是个什么数？”

“这是我接受你的征聘启事中提到的工作时人家给我的。只有这件衬衫适合我穿。”

“工作？什么——呵，我想起来了。转过身来。”蓝妖怪的运动衫的背面是“JUNIOR SOULS ASSISTANT”（初级精灵助理）。“好，Junior，让我瞧瞧你有多能干。出去跑一个，我给你传一个球。”Junior



Souls Assistant 沿着岸边快步跑了起来。Beast 把它那骨棱分明的手臂向后扬,接着扔出一个椭圆形的家伙飞向 Junior 的脑袋。Junior 伸手想抓住那东西,却没有抓住,眼睁睁地看着它从地上弹起来飞进湖里。这椭圆形的东西接触湖面时尖叫起来,随后就沉下去不见了。

"对不起,我眼睛里进了些硫磺",Junior 说道,盯着在燃烧的湖上扩展开来的波纹。"这东西是个好的吧?"随后他发觉自己讲错而一下住了嘴。"真是个愚蠢的问题。我们才不会带好的东西到这儿。"

"它连个差劲的都算不上",Beast 说。"只不过是个不值钱的橡皮精灵,连皮革都不是。但你无须担忧丢掉了这个精灵,我这里什么时候都有多得很的有罪精灵供你消遣。"他顺手从一大堆东西中捡起另一个仔细察看。"呵,是'她'",Beast 轻率地说。"不错,确实她注定要死在这里。"不过,他指着用歪歪斜斜的黑体字刻印在这精灵上的名字接着说道:"至少她知道她究竟是谁。而我就不一样了。"他的眼泪突然一下子涌了上来。

Junior 的蓝色变浅了一些。所有人都知道,Beast 正在经历一场死亡中期的身份危机。这个庞大的绿色妖怪突然一下跳起来,开始乱踢东西。"我是谁?"他咆哮大叫。"几千年了,没有任何人告诉过我究竟是谁?"然后他嚎啕大哭起来。

"我听说在阿拉米语中——阿拉米语是《启示录》最初所用的语言——666 的符号拼读起来是'尼禄'(Nero)。你想过你或许就是尼禄吗?"

"'罗马熊熊燃烧,尼禄放荡依旧……'罗马城烧不烧掉我倒不在乎,不过你知道我也不是一个出色的浪荡子。"

"嗯,耶稣会士 Father Bongus 把你的数字解释为'马丁·路德'(Martin Luther)",Junior Souls Assistant 大胆说道,用手同情地拍了拍 Beast 的两只角之间的额头。"他采用了一个被称为 gematria 的



系统，其中 A=1，B=2……Z=26。"

"我知道这些。不过，另一方面德国数学家 Michael Stifel 却'证明'我就是教皇 Leo 十世"，Beast 抽泣着说。"他从'Leo Decimus'开始，只留下'LD CIMV'这几个字母，把其他字母全去掉了。"

"为什么？"

"因为留下来的这些字母是对应于罗马数字的字母。它们加起来等于 1 656，于是 Stifel 再加上一个 X(10)，因为开始时用的是 Leo X，并减去 M(1 000)，因为这是'mystery'的首字母。"Beast 做了个鬼脸。"你听说过这样愚蠢的论证吗？"

"从来没有听到过"，Junior 说。"很明显这是生拉硬扯，牵强附会。"

"完全正确。每个人都有自己的一套想法。更糟的是"，Beast 继续说道，"他们全都用不同的规则来给字母赋与数值。没有任何人问一下是否存在一种不依赖于字母次序或其他任意选择的比较合理的方法。"

"你提到这点真是妙极了。我刚读了 1990 年 2 月号的'文字游戏'杂志(Word Ways)。"

"这是本什么杂志？"

"嗯，是本关于游戏语言学的杂志。"

"那就麻烦了"，Beast 说，"本文讲的是数学游戏。"

"呵，但那一期上有篇把数学游戏和语言学游戏融为一体的文章，是数学文字游戏专家 Lee Sallow 写的。该文被称为'新命理学'。Sallow 指出，'把每个字母同其位置序数联系起来的这一历史悠久的做法是一项可以抛弃的约定，因为它没有什么用处。新命理学以此作为它的出发点。'"

"我听不懂你说些什么"，Beast 烦躁地打断了他的话。

"嗯，我们以英语单词'one'(1)为例。按照英语中通常的字母排



序，这个单词的数值——或 gematric 常数——是 15 + 14 + 5 = 34。但是，如果命理学有什么内在意义的话，那这个 gematric 常数很明显应当等于这个单词所表示的数，即 1。而实际上并非如此。"

"有没有什么数字-单词的总数等于它们的 gematric 常数呢？"

"据 Dave Morice 说，没有——至少在英语中没有。除非你数一下这类短语'This is a beastly text: numerological constant of six-six-six。'"(此句中各单词的 gematric 常数之总和等于 666)

"那么这个叫 Sallows 的家伙主张怎么办呢？"

"给每个字母赋与不按其在字母表中的位置决定的数值。如果按照这些赋值一个数字-单词的 gematric 常数等于其数值，我们就说这个数字-单词是完全的。然后我们来试一下使尽可能多的诸如 ONE(1)、TWO(2)之类的数字-单词成为完全的。Sallows 把注意力集中于整数值上，并规定了一条限制，即不同的字母必须赋予不同的数值。"

Beast 使劲擤了一下他的鼻子。"这是什么意思呢？"他吸了一口气说。

"嗯，你得到一整套如下形式的方程：

$O + N + E = 1$

$T + W + O = 2$

$T + H + R + E + E = 3$

其中 O、N、E、T、W、H、R 等是代数未知数。看一下这些方程中有多少可以解出来，然后对答案进行调整，使它们能满足你打算规定的其他任何规则。例如，从方程 $O + N + E = 1$ 你可以看出，某些数必须为负数。此外，由于 N 和 E 同时出现于 NINE 和 TEN 中，因此我们有理由假定 N 和 E 已经被赋值，然后再看看会发生什么情况。例如，这样一来 O 就必须满足 $O = 1 - N - E$。由 NINE 和 TEN，可得 $I = 9 - 2N - E$，$T = 10 - N - E$。再次，由于我们希望 $T + W + O =$

2,我们发现 $W=2-O-T=2-(1-N-E)-(10-N-E)=-9+2N+2E$。

“不过你得当心。假定你决定由 $E=4$,$N=2$ 开始。这样可算出 T 也为 4——同 E 一样。由于不允许两个字母的值相等,因此你得为 E 和 N 选择另外的数值。”

Beast 一下子振作起来。“我明白了! 假定你尝试一下 $E=1$,$N=2$。这样就必定得到 $I=4$,$T=7$,$O=-2$ 和 $W=-3$。如果我们希望 THREE(3)也是完全的,那又如何呢? 嗯,这里有两个新字母,即 H 和 R。如果我猜测 $H=3$ 并依据 $T+H+R+E+E=3$ 这一事实,则 R 必须为 -9。接下来是 FOUR(4),它又有两个新字母,即 F 和 U。如果我选定 $F=5$,则由 $F+O+U+R=4$ 可得 $U=10$。看来我们搞得还不错,Junior。”

“完全正确。然后由 $F+I+V+E=5$ 可得 $V=-5$。由于 SIX(6)有两个新字母,我们先来解决 SEVEN(7)以确定 S 的值。由 $S+E+V+E+N=7$,得 $S=8$。然后我们可根据 $S+I+X=6$ 求出 $X=-6$。接下来再看 EIGHT(8)的方程,得出 $G=-7$。”

“这样,从 ONE(1)到 TEN(10)的所有数字名称现在都是完全的了。我们还能更上一层楼吗?”Beast 问道。

“或许可以吧。ELEVEN(11)和 TWELVE(12)中唯一的新字母是 L。如果 L 可以选择得使这两个数字名称同时成为完全的,那就实在是太凑巧了。”

“不错,但是事实上 $L=11$ 的确使得它们同时成为完全的。”

“妙极了。现在 $T+H+I+R+T+E+E+E+N=7+3+4+(-9)+7+1+1+1+2$,得 16。啊,天哪!”

“我想我们如果改变一下前面选择的数或许就可以走得更远些”,Junior 指出。“有几个数值完全是随意选定的。”

“我不敢肯定这样做能解决问题”,Beast 反驳道。“瞧,如果取下



面这个方程：

THREE(3)+TEN(10)=THIRTEEN(13)并把方程两边同时出现的字母对消，这样最终得到E=I。而这就违背了不同的字母取不同数值的规定。”

Junior Souls Assistant 点点头。“现在我想起来了，Sallows 曾经发现过这一论据。他说这是一个新的命理学的证据，证明13是不吉利的。”

4	17	2	16
24	9	20	6
25	12	22	7
1	27	11	3

图　Lee Sallows 的数字拼读游戏。在棋盘上任选一数，把它一个字母一个字母地拼读出来并把对应的数字加起来（减去黑色方格内的数字，加上白色方格内的数字）。所得结果始终等于你所选定的那个数字（或其相反数）

“我们还可以朝反方向发展。看看这个式子：如果Z+E+R+O=0，则Z=10。”

“和U相同。真烦人。”

“是的，但我们曾对字母的值作过许多假设。或许改变这些假设可以使问题得到解决。”结果他们发现这样真的能解决问题（见英语

中的完全的数字-单词)。

"Sallows 介绍了以类似想法为基础的几项魔术",Junior 提出。例如,如果采用另一种赋值方式(如下所示),则从 ZERO(0)到 TWELVE(12)的所有数都是完全的,而且 FOURTEEN SIXTEEN,SEVENTEEN 和 NINETEEN 也是完全的:

E=0　I=1　R=5　V=14

F=−10　L=−7　S=−11　W=−1

G=9　N=4　T=6　X=16

H=−8　O=−3　U=12　Z=−2

Sallwos 的想法是准备一套卡片,每张卡片上有一个字母及其相应的数字。准备三张 E/O 片卡,两张 N/4 片卡,其余每个字母为一张卡片,总共 19 张卡片。请某位观众拼读出一个数字单词来,并把卡片上对应的数字相加,所得结果就刚好等于这个单词表示的数字,除了符号以外——所得结果可以是正的,也可以是负的。"

"如果某个聪明的家伙要拼读 THIRTEEN(13)或 FIFTEEN(15)又怎么样呢?"Beast 问道。

"这是不可能的。因为卡片中只有一个 T 和一个 F。"

"真是鬼得要命。Sallows 这家伙很适合在这儿。"

"不错,他发表了一个使用 4×4 棋盘来玩的类似把戏(见图)。在棋盘上任选一数,把这个数拼读出来,并把各个字母所对应的数字相加。白色方格的数字为正,黑色方格的数字为负。这样相加的结果就是你选定的那个数。"

Beast 大笑起来,其笑声有点令人毛骨悚然。他兴奋地一连把 12 个有罪精灵踢到地平线外看不见的地方,随后又把第 13 个精灵拍来拍去,这精灵一触到地面就发出低沉的嚎叫声。跟着 Beast 的脸突然沉了下来。

"怎么啦,Beast 先生?"



“用英语名称来搞结果倒是挺不错的。但是用法语又如何呢?”

“噢,是的。这是一类完全新的问题。在法语中数字单词是 UN(1),DEUX(2),TROIS(3),QUATRE(4),CINQ(5),SIX(6),SEPT(7),HUIT(8),NEUF(9),DIX(10),ONZE(11),DOUZE(12),TREIZE(13),QUATORZE(14)等等。你可以一直搞到 13,但 14 就不行了。因为把等式 QUATRE + ONZE = UN + QUATORZE 两边的相同的字母对消后,就得 E = U,但由于不同的字母必须取不同的数值,因此 E = U 是不允许的。在法语中这整个游戏是相当严格的。如果你按正确的次序考虑从 ZERO(0)到 TREIZE(13)这些数字单词,你将发现有 11 个字母完全由 N 的值所决定;而除了 N 以外,另外就只有一个字母(A)的值能自由选择了。这就是说,你必须按下列各式决定定母的值:

A = *　　　　P = 2

C = A − 5N − 4　Q = 2N + 5 − A

D = 2N　　R = N − 11

E = 3N − 5　　S = 2N − 4

F = 13 − 3N　　T = 14 − 5N

H = 4N − 11　　U = 1 − N

I = 2N + 4　　X = 6 − 4N

N = *　　　Z = 16 − 4N

O = 0

如果取 A = 20,N = 7,那么所有字母的数值就各不相同,而且也不大。最终得到的赋值情况是:

A = 20　H = 17　Q = −1　X = −22

C = −19　I = 18　R = −4　Z = −12

D = 14　N = 7　S = 10

E = 16　O = 0　T = −21



F = −8　P = 2　U = −6

这样从 ZERO(0)到 TREIZE(13)的每个数字单词都是完全的。"

"德语又如何呢?"

"唔。Sallows 在他的文章中没有提到德语。我看我们最好是自己把它搞出来。"

"我先来试试。如果不算零——因为我觉得这样会好些——德语的数字单词是 EINS(1),ZWEI(2),DREI(3),VIER(4),FÜNF(5),SECHS(6),SIEBEN(7),ACHT(8),NEUN(9),ZEHN(10),ELF(11),ZWOLF(12),DREIZEHN(13),VIERZEHN(14),FÜNFZEHN(15),SECHSZEHN(16),SIEBZEHN(17),ACHTZEHN(18),NEUNZEHN(19),ZWANZIG(20)。我想我们最好把 Ü 看做和 U 是一样的。"

"就这样,Beast 先生。嗨,我觉得我们看出点名堂来了。瞧,从德语中 13 到 19 的构词方式可以看出,如果你能解决从 1 到 10 的数字单词,那么从 13 到 19 也就自动解决了。"

"除了 SIEBZEHN(17)之外。德语中的 17 不是 SIEBENZEHN。"

"不错。但这一点告诉我们,由于 SIEBEN + ZEHN = SIEBZEHN,把等式两边相同的字母对消后,必须有 E + N = 0。然后由 E + I + N + S = 1 可得 I + S = 1。假定我们令 E 和 I 取我们希望取的任何数值,这样就必定有:

E = *

I = *

N = −E

S = 1 − I

接下来是 ZWEI(2)。我们可让 Z 取任意值,从而推导出:



Z = *

W = 2 − E − I − Z

依此类推。情况逐渐复杂起来，但我想如果我们系统地进行考虑，则就可能——"

"是的。我刚想到一个主意。德语中的'21'是 EINUNDZWANZIG，如此等等。如果我们设法使 UND 各字母的总和为零，则我们不费吹灰之力就可解决从 21 到 29 的各个数。"Junior 和 Beast 一连几个小时用他们的尾巴尖在地狱的硫磺沙地上写画着。最后他们得到了以下结果：

A = −10　F = −2　N = 1　V = 8

B = 7　G = 33　O = −6　V = −8

C = −18　H = 17　R = 16　W = 13

D = 9　I = −3　S = 4　Z = −7

E = −1　L = 14　T = 19

这样从 EINS(1)到 NEUNUNDZWANZIG(29)的所有数字名称都是完全的了。

"干得不错，Beast 先生，"Junior 大声说道。"我想知道我们是否也能把 DREIZIG(30)也弄成完全的？如果成功的话，那么从 31 到 39 的所有数也就自动解决了。此外，意大利语、西班牙语、俄语、希腊语、日语、洋泾滨英语等等又怎样呢？"

"我认为，最好是让人间的精灵去为此伤脑筋吧"，Beast 哼着鼻子说道。

英语中的完全的数字-单词

指定数值

E = 3　I = −4　R = −6　V = −3

F = 9　L = 0　S = −1　W = 7

G = 6　N = 5　T = 2　X = 11



H=1　O=-7　U=8　Z=10

那么

Z+E+R+O=0　F+I+V+E=5　N+I+N+E=9

O+N+E=1　S+I+X=6　T+E+N=10

T+W+O=2　S+E+V+E+N=7　E+L+E+V+E+N=11

T+H+R+E+E=3　E+I+G+H+T=8　T+W+E+L+V+E=12

F+O+U+R=4

22. 费马的最后旅行

6 月下旬的一个暖意融融的晚上，我坐在花园的躺椅上读报。一条头版消息报道说普林斯顿大学的 A.J. 威尔斯已解决了数学中的重大问题之一——费马的最后定理。威尔斯在三次讲学结束时宣布了这项成果，这些讲学的题目看来是很单纯的："模形式、椭圆曲线和伽罗瓦表示"。济济一堂的听众猜想威尔斯早已成竹在胸，他们的这一猜测是正确的。我很想知道如果法国数学家皮埃尔·德·费马突然一下从 350 年前跑到未来，他会如何对待这一切。夜晚的空气仍然非常温暖，于是我决定合上眼睛小睡一下。我没有睡着很久就被一阵新奇的呜呜尖叫声所弄醒，抬头一看，见草地上出现一个由闪闪发光的金属和玻璃制成的奇妙东西。有个人坐在这装置里。他穿着黑色长袍，长长的头发用一根带子扎了起来。他从装置中爬出来，自我介绍说他就是皮埃尔·德·费马。"对不起"，我结结巴巴地说，"我以为你 300 多年前就去世了。"

"嗯，决无此事。我刚才遇见了这位有趣的绅士 H.G. 威尔斯——或许你认识他。"



“我知道他”,我用一种要窒息了的声调说道。

“他宣称他来自未来,我不相信会有这种事,所以他把他的这台装置借给了我。现在我明白他说的是真话。”

“你来得正是时候”,我告诉他,“你的最后定理刚刚被证明了。”

“我的什么东西?”

“你还记得你在你的《算术》一书的页边上所写下的一个注吗?你当时写道:‘把一个立方分解为两个立方之和,一个四次幂分解为两个四次幂之和,或一般地说把一个更高次幂分解为两个相同次幂之和是不可能的。关于这一事实,我已发现了一个引人注目的证明,不过这页边太小了,写不下。’”

“不错,确有此事。”

“如果用代数符号来表示,你所说的就是,如果 n 为大于 2 的整数,且 x,y,z 为正整数,则方程 $x^n+y^n=z^n$ 无解。这一猜测后来被称为费马的最后定理,因为许多年来它都是你提出的所有命题中剩下的唯一一个既未被后来的数学家证明,亦未被证否的命题。没有任何人能重构出你提到的所谓‘引人注目的证明’,而且大多数人都怀疑你是否真的有过这样的证明。300 多年来,世界上最优秀的数学家千方百计想攻克你的最后定理——有时获得一些进展,但从未完全证明它——直至 A. 威尔斯戏剧性地解决了这个问题。”

我把报纸递给费马看。在他读这篇文章时,我在一旁等待着。“人们称这个定理为数学的‘神圣的梦想’?‘世纪定理’?天哪!”

“你有证明吗?”我问道。

费马微笑了。“别着急。在我抖出我的秘密之前,我得先请你告诉我数学上那些最终导致我的平凡猜想得以证明的重大进展。”

“可以”,我说,“但是首先你必须让我知道你是如何提出你的最后定理的。”

“好吧。我的首要爱好始终是数论,也就是研究普通整数的数学



分支。你也许知道,一位名叫丢番图的希腊人发明了'丢番图方程'这一概念——这种方程只允许有整数解——并写了一本题为《算术》的书来介绍它,从而开创了数论这个领域。一个重要的例子是毕达哥拉斯方程,即 $x^2+y^2=z^2$:两个二次幂之和仍为一个二次幂。此方程有整数解,如 $3^2+4^2=5^2$ 和 $5^2+12^2=13^2$ 等。事实上,它有无穷多个整数解。

"在1637年的某个时候,我读了丢番图的著作并思考了毕达哥拉斯方程。我想到这样一个问题:如果不用二次幂(平方)而用三次幂(立方),那将会出现什么情况呢?首先我验算了某些简单的情况:1^3+2^3 是一个三次幂吗?不是,它等于9。2^3+3^3 是一个三次幂吗?不是。后来我几乎成功:例如,$9^3+10^3=1\ 729$,而 $12^3=1\ 728$,但是,除了 $0^3+1^3=1^3$ 之类的毫无意思的情况外(此时有一个数为零),我没有发现这个方程的任何整数解。由于在寻找此方程的解时一再碰壁,我想到了我的那个朴素的猜想。"

"好,现在让我谈谈你的猜想是如何证明的",我说道,"你知道,我们只需要对 $n=4$ 和 n 为素数这两种情况证明该定理就行了。"

"不错。这是因为比如说每个完全的15次幂同时也是一个完全的三次幂,因此对15次幂的任何解同时也就是三次幂方程的一个解。"

"的确如此。用符号来表示就是 $x^{15}=(x^5)^3$,因此,如果 $x^{15}+y^{15}=z^{15}$,那么 $(x^5)^3+(y^5)^3=(z^5)^3$。由于任何大于2的整数或者可被4整除,或者可被一个奇素数整除,因而运用同样的推理可以证明4和奇素数幂是唯一需要证明的两种情况。你自己走出了第一步,即提出了当 $n=4$(四次幂)时的证明。"

"是的,我对此感到自豪——我用的是'无穷递降法'。出于技巧上的原因,我假定某种稍微更具一般性的方程 $x^4+y^4=z^2$ 有一个解。说它更具一般性,是因为任何四次幂同时也是一个二次幂。我注意

到这样一个解将得出一种其两边本身为二次幂的毕达哥拉斯三角形(见毕达哥拉斯三角形)。我运用毕达哥拉斯三角形的标准公式,作了几次简单的推导即发现可以构造出方程 $x^4+y^4=z^2$ 的另一个解,此解中的 x,y 和 z 具有更小的非零值。照这种方法推下去,我发现必然会得到这样一个结论,即这个方程只要存在任何解,就必定可以推导出最小的非零整数——即 $x=1,y=1$——也是这个方程的解。然而实际上它并不是该方程的解。因此,方程 $x^4+y^4=z^2$ 不可能存在整数解,特别是方程 $x^4+y^4=z^4$ 不可能存在整数解。"

"妙极了",我说,"这样,唯一留下来尚待解决的就是 n 为奇素数的情形了。你也证明了 $n=3$,即两个三次幂之和的情况。瑞士数学家 L.欧拉独立地证明了 $n=3$ 和 $n=4$ 这两种情况。1828 年,P.G.L.狄里赫莱证明了 $n=5$ 的情况,A.M.勒让德则在 1830 年给出了其证明。1839 年,G.拉梅试图证明 $n=7$ 的情况,但他犯了若干错误,而 L.勒贝格则在 1840 年纠正了他的错误。"

"这么说来,过了两个世纪后只证明了 $n=3,4,5,7$ 这几种特殊情况吗?难道没有人想出更一般的办法吗?"

"拉梅在 1847 年提出了较一般的设想。他宣称有了一个对所有 n 次幂的证明方法。但是 E.E.库麦尔发现了其中有一个错误——一个非常有趣的、为未来的进展指明道路的错误。拉梅的基本战略被证明是富有成果的,但他的战术却很不高明。"

"说说他的基本设想好吗?"

"就是引入一些被称作代数数的新数。比起整数来,代数数属于更一般的数。之所以被称作代数数是因为它们是代数方程的解,但细节在这里并不重要。表达式 x^n+y^n 可以写成另外两个表达式之积。例如,当 $n=5$ 时,有:

$$x^5+y^5=(x+y)(x^4+x^3y+x^2y^2+xy^3+y^4)$$

$x+y$ 这个因式简洁而漂亮,但第二个因子就复杂多了。拉梅注



意到，运用他的代数数，可以把这个复杂因子表示为四个简洁得多的因子之积。更一般地说，他可以把 X^n+Y^n 表为 n 个简单项之积。此外，由于 $x^n+y^n=z^n$，这些项之积是一个完全的 n 次幂。他还注意到，这些项中任何两项均无公约数。对于整数来说，如果各项之间无公约数的若干项之积为 n 次幂，则每一项分别也为一 n 次幂。这一命题仅依据于这样一个事实，即每个整数只能唯一地表示为若干素因子之积。

"拉梅假定这一性质也适用于代数数。然后，他用 n 个不同的方程——每个方程表示 n 项中的一项是一个 n 次幂——来代替你的那个简单的方程。而且所有的方程必须同时成立。这一要求是很高的，毋庸奇怪，拉梅得以证明了没有解存在。"

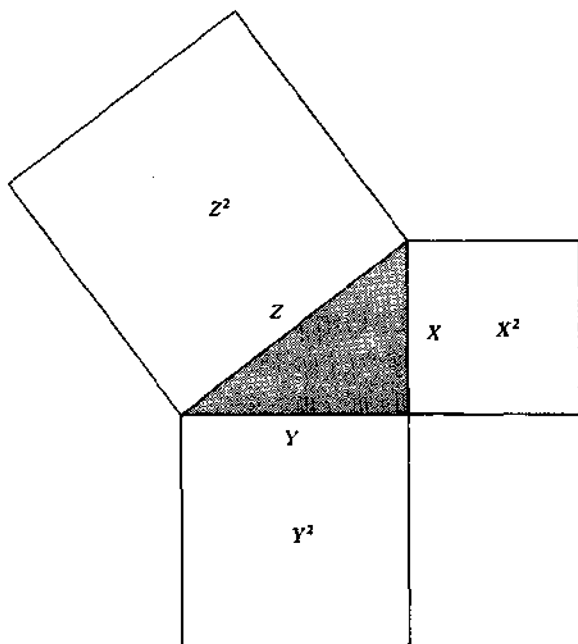
"嗯，"费马说，"我有某些类似的想法。不过——"

"事情并不那么简单，对吧？"我打断了他的话头。"库麦尔和其他一些人指出了对于 n=23，拉梅的代数数表示为素数之积的方法不止一种。这是非常奇怪的，其计算也是有点烦人的——如果你愿意的话，我可以写给你看。"

他挥手让我说下去。"不用了。我自己去考虑这个问题。真是迷人。"

"库麦尔提出一个问题：为什么代数数可以有一种以上的素因子分解法。最后他发现，引入一种他称为理想数的完全新的东西，整个问题便迎刃而解。理想数提供了某些'额外'的素因子，从而使一切都归于正常。你可以看出这个问题现在开始有点复杂并且越来越抽象了，不过这是数学不可避免的发展道路。

"到 1847 年，库麦尔已经运用他的理想数理论对于 n 为 100 以下的所有情形(除了 n=37，59 和 67 这三种情况外)证明了你的猜想。通过向这一数学方法添加若干额外的手段，库麦尔和 D. 米利曼诺夫在 1857 年也解决了这三种情况。运用类似的方法到 1992 年



已证明了 n 等于或小于 100 万时的所有情形。”

“但是几乎所有的数都大于 100 万”，费马指出，“这样一种逐个证明的办法永远也不能彻底解决问题。”

“的确如此。当然，在任何时候某个人或许会注意到能够揭开整个问题的某一奥妙。但是事情并不是如此发展的。我们只是看到越来越复杂的特殊情形。需要开辟新的思路。而新的思路是沿着一条相当不同的途径来到的。人们开始思考一个丢番图方程究竟可能有多少解。有的丢番图方程有无穷多个解，如毕达哥拉斯方程。有的丢番图方程无解，比如当 $3 \leqslant n \leqslant 100$ 万时你的那个方程就是这样（不考虑平凡解）。有的方程则有有限多个解，如 $y^2+2=x^3$，它的正整数解只有一组，即 $x=3, y=5$。



"1992 年,英国数学家 Louis J. 莫德尔试图确定上述各种可能情况由什么因素决定,而且他逐渐察觉到了一种可能的规律。他注意到,如果考虑这样一个方程的所有复数解——也就是把解集扩大到最一般的地步,不作任何关于整数解的假设——那么这些解构成一个拓扑表面。这个表面有有限个'洞',就像面包圈或椒盐卷饼那样。他觉得非常引人注目的是,有无限多个整数解的方程如果在复数集上求解,则总是没有洞或只有一个洞。拓扑学与算术之间似乎存在一种联系。"

"这真是个大胆的主意——没有人能够想出任何办法来得出这两个截然不同的数学分支之间的牢固联系。但是莫德尔对此深信不疑,因此他发表了一个猜想(现在被称为莫德尔猜想),其内容是说给出有两个或两个以上的洞的表面的方程仅有有限多个整数解。"

费马看来有点迷惑不解。"这跟我的猜想有什么关系?"

"对应于你的方程 $x^n+y^n=z^n$ 的表面上的洞的数目为 $(n-1)(n-2)/2$,当 $n>3$ 时,此数至少为 2。这样从莫德尔猜想就可以推出,如果你的方程有整数解的话,则它必定只有有限多个整数解。"

费马似乎更糊涂了。"但是,如果 X、Y 和 Z 组成了一个解的话,那么 2X,2Y 和 2Z,3X,3Y 和 3Z 等等也都是解,这样的解有无穷多个。"

"呵。我本该说清楚莫德尔指的是没有任何公因子的解。"

"这下我明白了。"

"很好。接着出现了第一个重大突破。在 1962 年,I.R. 萨发列维奇提出了关于丢番图方程的一个新的专业性很强的猜想。1968 年,A.N. 帕辛证明了萨发列维奇的猜想蕴含了莫德尔猜想。最后,在 1983 年,年轻的德国数学家 G. 法尔丁斯证明了帕辛的猜想,从而也就证明了莫德尔猜想。这就意味着你的猜想是几乎为真的:如果对于任何 n 出现了例外情况,那么它们的数目也是有限的。你或



许会喜欢法尔丁斯的证明:它采用了你的无穷递降法的一个变形。"

"啊。"

"但却用在一类被称为阿贝尔簇的非常抽象的对象上。"

"哦。不过,看到我的那个简单猜想竟产生了如此之多深刻而强有力的数学新概念,真是令人感到欣慰。"

"的确如此。好了,你可能会提出异议说,有限多个解与无解并不是一回事。这是完全正确的。但是你将意识到,从可能有无穷多个解简化到只有有限多个解是一个巨大的进展。之后不久,D.R.希斯-布朗改进了法尔丁斯的方法,证明了当n非常之大时,使你的猜想成立的整数n所占的比例趋近100%。你的最后定理是'几乎恒为真'。"

费马显得很高兴。"我觉得这是一个普遍的结果,但或许不大具体,因为它没有指出哪些n值是例外情况。"

"不错。还缺乏一个更具体的设想。这一设想来自于一个构成丢番图方程的现代研究方法的核心的极为精美的理论,就是椭圆曲线的理论。"

"什么是椭圆曲线?"

"形如$y^2=ax^3+bx^2+cx+d$的方程——即一个完全平方等于一个三次多项式——所表示的曲线。它们之所以被称为'椭圆曲线',其'椭圆'一词的来源是因为它们与寻找椭圆周长的公式的问题有某种模糊的关系,其'曲线'一词的来源则是因为每个方程都通过坐标几何定义了一条几何曲线。椭圆曲线的一个引人注目的特征是,只要给出该方程的几个整数解,就可以把它们组合起来得出其他解。有一种几何作图法可以用来从已知解构造出新的解(见椭圆曲线上的点)。

"椭圆曲线是启发莫德尔得出他的猜想的因素之一,因为与椭圆曲线相关的曲面只有一个洞或没有洞。数学家们在若干年的时间里



建立起了一个非常强有力的椭圆曲线理论。你可以说它们是丢番图方程的唯一一个被众充分掌握了的领域。但是这个领域也存在它自己的尚未被解决的重大问题，其中最大的一个是所谓谷山-魏尔猜想。这一定理是说任何一条椭圆曲线均可用模函数表示(模函数是通常的三角函数——如正弦、余弦等——的一种广义形式)。它意味着每条椭圆曲线都是某种精美的坐标系统。

"现在我们进入最后冲刺的阶段了。80 年代初，萨尔州大学的 G. 弗雷确立了你的最后定理和椭圆曲线之间的一个关键联系。他把数论中理解最少的领域和认识得最充分的领域联系起来了。弗雷的思路是这样的：假定你的方程有一个解为 $X^n + Y^n = Z^n$。这里 X, Y, Z 用大写字母是为了表示我们考虑的是某个具体的解。你想要证明没有这样的解存在，这样你只需假定有这样一个解存在，然后一直推导下去直至得出某个矛盾的结论。矛盾在什么地方是无关紧要的。"

"归谬法"，费马指出。

"现在我们称其为'反证法'，不过反正就是这么回事。弗雷遵照巴黎的法兰西学院的 J.P. 西雷的建议，考察了椭圆曲线 $y^2 = x(x - X^n)(x - Y^n)$。他采用椭圆曲线的一般理论来研究它，发现了现在被称为弗雷椭圆曲线的曲线是非常奇异的东西。它具有各种性质的奇特组合，奇特到这样一种怪物的存在看来是非常不可能的，而这自然正是你想要证明的东西。1986 年，伯克利加利福尼亚大学的 K.A. 里伯特使弗雷的设想得以精确化。他证明了如果谷山-魏尔猜想成立，则弗雷的椭圆曲线不可能存在。这将通过反证彻底证明你的猜想。"

"这是一个极其重大的发现。说明你的猜想不是单单一个孤独的新鲜玩意。相反，它与现代数论的核心——谷山-魏尔猜想有关。A. 威尔斯在童年时就想要证明费马的最后定理。当他成为专业数



学家后，他断定这个定理只是一个孤立的困难问题——证明它是件好事，可是除了名声很大外它并没有什么真正的重要性。但当他得知里伯特的研究工作后，他决定把自己的全部研究工作都投入到证明费马定理上。他意识到，要使这一方法行得通，无须借助谷山-魏尔猜想的全部力量；你只需要它的适用于所谓半稳定椭圆曲线的一个特例就行了。威尔斯把这个问题分为 6 个部分，并逐个地证明出来直至只剩一个未被证明。后来，哈佛大学的 B.C. 马楚尔关于另一个完全不同的问题的演讲激发起了新的设想，为威尔斯提供了最后的线索。在一篇长 200 页的论文中，威尔斯动用了足够的数学方法证明了谷山-魏尔猜想的这一特例。这样威尔斯证明了谷山-魏尔猜想的半稳定情形，而这又意味着里伯特的论证证明了弗雷的椭圆曲线是不存在的——而你则始终是正确的。”我深深地吸了一口气。“威尔斯有许许多多强有力的数学工具可用，但他用了整整 7 年的艰苦努力才弄清如何把各部分拼凑成一个整体。他知道总的战略，但必须把他的战术搞对头。这是一个惊人的成果。”

费马谨慎地点点头。“这一证明经其他人验证过了吗？我从我自己的研究工作中知道出错是很容易的——”

“还没有在所有细节上验证过。但是人数多得惊人的专家现在都愿意说他们相信这一证明。马楚尔总结了大家的共同看法：‘威尔斯的证明听起来像是真的。’”

我盯着费马。“我已完成我该做的事了。现在轮到你了。你真的得到过一个证明吗？就是你说页边空白太小写不下的那个证明。我们是否需要所有这些精美而复杂的数学工具呢？抑或是仍然存在一个有待发现的简单证明？”

“嗯”，费马开始说道，“事情是这样的。我——”正在这时，时间机器忽然抖动起来，一会儿消失一会儿又重新出现。“喔”，费马叫了起来，“时间旅行机对我提出警告了。我必须立刻回到我自己的时间



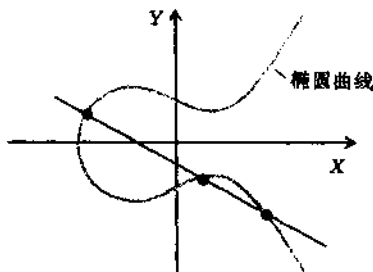
中，否则我就得永远呆在这里了。再见！”我还没来得及拦住他，他就登上了时间机器，按动操纵杆并消失了。

毕达哥拉斯三角形

为了解出毕达哥拉斯方程 $x^2+y^2=z^2$，取任意整数 K，U，V。令 $x=k(u^2-v^2)$，$y=2kuv$ 及 $z=k(u^2+v^2)$。由此可得该方程的一组解。例如，设 $k=1$，$u=2$ 及 $v=1$，则 $x=3$，$y=4$，$z=5$。或设 $k=1$，$u=3$，$v=2$，则 $x=5$，$y=12$，$z=13$。此法可产生该方程所有的解。

椭圆曲线上的点

椭圆曲线上的点



一条直线与典型的椭圆曲线相交于 3 点。如果其中两点的坐标对应于连带丢番图方程的整数解，则第三点的坐标也对应于该方程的整数解。要从已有的解中构造出新解，取该方程的两个解，作一条通过相应两点的直线并计算出此直线与该曲线相交的第三点的坐标。

23. 哪种投票制度最合理

Blockvotia 的国民议会刚统计完了对 Palmgreasing Slushfund 议案的投票情况，总统 Freebie Perks 满脸的不高兴。他的私人秘书 Penelope Poundpincher 施展浑身解数拼命安慰他。

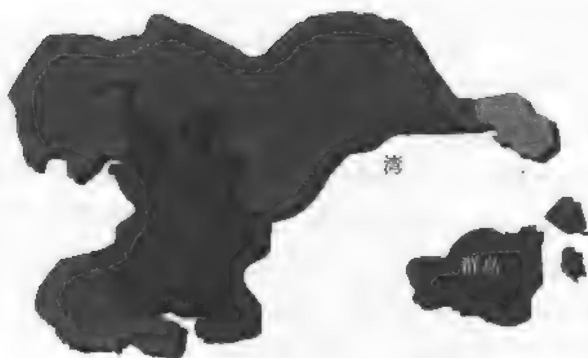
“Penny，你曾告诉我 6 个地区中有 4 个地区(包括最大的那个地区)支持此议案。那我们怎么输了呢?”

“这是由于加权投票制度的缘故，先生。你知道，每个地区都分配了与其人口数大致成比例的一定票数。这是说明详细分配情况的一张表(见图 1)。总的票数为 31。因此，任何一个拥有 16 张票(即比总票数的一半多一张票)的联盟将能够左右选举的结果。

Sheepshire ，Fiddlesex，Slurrey 和 Porkney 群岛对议案投了赞成票。我已经说过，这正好是 6 个地区中的 4 个，而且包括了最大的一个。但是它们合起来仅有 15 票，而反对议案的两个地区却有 16 票。”

“总统选举下个月就要进行，我不希望这一情况重演。如果我们让边界委员会给 Sheepshire 加一票，而使 Candlewick 减一票——”





Blockvotia 各地区的投票权重		修改后的投票权重		第三次选择的投票权重	
地区	票数	地区	票数	地区	票数
	10		10		12
	9		9		9
	7		7		7
	3		3		3
	1		2		1
群岛	2	群岛	2	群岛	1

图 1　Blockvotia 的地图示出了各地区的大小。地图下的表列出了现行的投票加权和另外两种可能性

Penny 把头摇得像拨浪鼓。“我不主张这样做，先生。Richfolk 和 Candkewick 这两个区都赞成你连任。Sheepshire 犹豫不决，而另外三个地区则反对你连任。Richfolk 和 Candlewick 可以挫败另外 4 个区组成的联盟，但如果你从这两个区中任一个区减去一票的话，那情况就不同了。”

这时有人敲门。Porkney 群岛的代表 Charlie Hogg 冲了进来。“总统先生，这场滑稽戏不能再继续演下去了！”

“什么滑稽戏?”



"你所谓的民主投票制度,Porkney 群岛毫无权力。"

"但你们拥有一票,这是与你们的人口成比例的。Slurrey 的人口比你们还多,也只有一票。你们拥有的权力实际上比 Slurrey 还多。"

"不对。任何投票的结果都完全被三个最大的地区所左右。这三个地区至少有两个将投相同的票,而它们合起来的票数至少同 Richfolk 和 Candlewick(分别是第二大和第三大的地区)拥有的总票数一样多。这就有了 16 票,居于多数地位。在任何一次投票中,即使 3 个最小的地区一票也不投,也会得到相同的结果!"

"我明白了。但我能把它怎么样呢?"

"再给我们一票!这样至少 3 个最小的地区就可以同 Sheepshire 联合起来打出一个平局。如果你再给 Slurrey 也加一票,那么我们就可以结成一个获胜的联盟了(见图 1)。"

"我明白你的意思。这样总票数就是 33",Penny 说。"这样有 17 票或 17 票以上就可以得胜。Fiddlesex,Slurrey,Porkney 群岛和 Sheepshire 结成联盟,能够赢得投票。"

"不错!三个最小的地区中的任何一个都能够改变投票的结果——它们将拥有力量均势!"

这时边界委员会的联络官 Gerry Mander 走了进来。Perks 问他:"Gerry,边界委员会能否重新划定各区的边界,使 Slurrey 和 Porkney 群岛各多得一票?"

Cerry Mander 摇了摇头。"对 Slurrey 区还可想点办法。但 Porkney 是群岛就不好办了。"

Hogg 咆哮起来:"我的选民们会不高兴的。"

总统叽咕着说:"是会不高兴。不过,正如你说的那样,这没有什么用,因为你的地区毫无权力。我看你最好不要发出无法兑现的威胁,Hogg。"

"单是三个选区就可以把你赶下台,这种情况也不会使你感到舒



服。你必定能够想点什么办法。"

"我可以再给 Sheepshire 两票。能办到吗,Gerry?"

"没问题。区界沿着 Wastedump 河弯弯曲曲地延伸。我们很容易把它改合理一些。"

"但是给最大的地区再加几票不可能帮助最小的地区获得一份权力呀!"Hogg 伤心地叫道。

Perks 说:"恰恰相反,如果 Sheepshire 再多两票的话,你就会得到一份权力(见图 1 的右图)。"

"不错",Penny 边说边看着这些数字,"这同一个联盟共拥有 33 票中的 17 票;依然是最小的三个地区中的每一个都可以声称自己掌握着力量均势。"

"这真是妙极了",Hogg 说,"你给了 Sheepshire 更多的权力,其中部分权力却鬼使神差般地影响到我们。"

"不,Hogg。我们并没有给 Sheepshire 更多的权力——我们只是给他们更多的选票",Penny 吸了一口气说,"正如你说的那样,权力和选票并不是一回事。"

"怎么会是这样?"Perks 问道,"如果权力不是选票,那它是什么?我需要知道这一点。权力赢得选举。"

"我认为你需要 Banzhaf 权力指数,先生",Penny 说,"John F. Banzhaf Ⅲ是乔治敦大学的一位法律专家。1965 年他提出了在加权投票体制中衡量代表所拥有的权力的一种方法。他的设想是,一位代表可以通过加入一个看来要输掉的联盟使其转败为胜。或者背弃一个看来要获胜的联盟使其转胜为败而显示其权力。"

"这不是同一回事吗?"

"是同一回事,先生。如果你加入一个联盟,你同时也就背弃了由其他所有人组成的另一个联盟。所以我们只需要考虑一种情况就够了——比如说考虑建立一个获胜的联盟。假定某一位代表在联盟



中起着关键性的作用：有了她则联盟赢得投票，失去她则联盟输掉投票。任何一位代表的 Banzhaf 权力指数就是她在其中恰好起着这样一种作用的联盟的数目。”

“我们原先的投票体制是一个(16;10,9,7,3,1,1)体制。获得多数所需的票数为 16 票。各个代表的加权为 10,9,7,3,1 和 1。Porkney 仅能在恰好有 16 票的联盟中起着关键作用。如果这种联盟有更多的选票，那么 Porkney 是否背弃它对投票结果不会有任何影响。如果其票数少于 16 票，则它就不是一个获胜联盟了。但是 Porkney 所属的任何一个联盟其选票总数均不等于 16 票，因此 Porkney 的权力指数为零。按照总统提出的新方案，我们将有一个(17;12,9,7,3,1,1)投票体制。Porkney 在其所属的任何一个恰好有 17 票的联盟中起着关键作用。这种联盟正好有一个，即由 Sheepshire, Fiddlesex, Slurrey 和 Porkney 组成的联盟。因此，Porkney 的权力指数为 2。”

“那么 Sheepshire 的权力指数为多少？”Perks 问。

“Sheepshire 有 12 票，因此它在它加入的任何一个拥有 17 票到 28 票(即 17 - 1 + 12 票)的联盟中起着关键作用。你可以通过试错法列出这些联盟(见图 2)。这种联盟共有 18 个，因此 Sheepshire 的权力指数为 18。”

Hogg 叫了起来：“Sheepshire 的人口是我们的人口的 12 倍，可他们的权力却只有我们的权力的 9 倍。”

Gerry 问道：“有没有比试错法更好的计算权力指数的方法呢？”

Penny 说：“嗯，对于大的投票体系，最好的办法是使用计算机。不过，对于小的投票体系(比如我们这一个)，有一种巧妙的图解法。假定此体系是(3;2,1,1)，这就是说，有三位投票人：A,B 和 C。A 有两票，B 和 C 各有一票，且 3 票构成多数。”

“首先画出一个显示出所有可能的联盟的点阵图；如果这些联盟



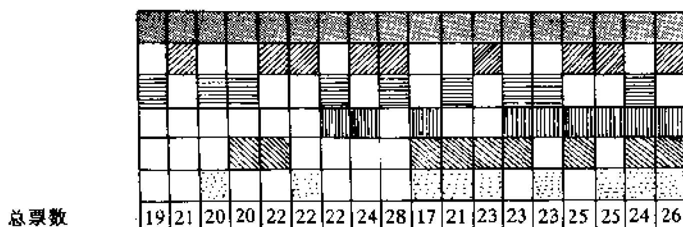


图 2　在上面这种投票体制中，Sheepshire 的权力指数为 18，这说明它在其中起着关键作用的联盟共有 18 个。理想的情况是，所有各地区的权力指数应当差不多相等，正如纽约州的汤普金斯县 1982 年的情形那样

仅相差一个成员，则把它们用一条边联接起来。在每条边上标以非两个联盟所共有的那个成员。然后标出每一条关键的边——也说是总票数从低于多数票变成等于多数票或高于多数票的那些边。任何一位成员的权力指数就是其上标有它的名字的那些边的数目。在这个例子中，A 出现在 3 条关键边上，因此它的权力指数为 3；B 和 C 则各出现在一条关键边上，因此其权力指数均为 1。这个点阵图是个立方体。对于更大的系统，你也可以画出点阵图，但是看起来就有点零乱了。不过四个成员的点阵图还是有点漂亮的（见图 3）。”

纽约州汤普金斯县议会（1982 年）

自治地区	人口	权重	权力指数	权力指数/人口
LANSING	8.317	404	4.747	0.571
DRYDEN EAST	7.604	333	4.402	0.579
ENFIELD & NEWFIELD	6.776	306	3.934	0.581
ITHACA WARD 3	6.550	398	3.806	0.581



续表

自治地区	人口	权重	权力指数	权力指数/人口
ITHACA WARD 4	6.002	374	3.474	0.579
ITHACA SOUTHEAST	5.932	470	3.418	0.576
ITHACA WARD 1	5.630	261	3.218	0.572
ITHACA WARD 2	5.378	246	3.094	0.575
ITHACANORTHEAST	5.235	241	3.022	0.577
GROTON	5.213	240	3.006	0.577
CAROLINE & DANBY	5.203	240	3.006	0.578
ITHACA AWRD 5	5.172	238	2.978	0.576
ITHACA WEST	4.855	224	2.798	0.576
ULYSSES	4.666	214	2.666	0.571
DRYDEN WEST	4.552	210	2.622	0.576

Hogg 说:"我希望的是每个成员拥有的权力指数大致同其人口成比例。"

"这可不那么容易办到",Penny 说,"让我向你说明纽约州汤普金斯县议会 1982 年是如何做到这一点的。权力指数几乎正好与人口成比例(见图 2)。"

"我们也可试试看",Hogg 提议说。

"或许可以吧",总统慢条斯理地说,"你对美国总统的权力指数作过研究吗,Penny?"

"是的,先生。美国总统的权力指数为一位参议员的权力指数的 40 倍,为一位众议员的权力指数的 175 倍。"

"这听起来很不错。"

"不过美国立法机构作为一个总体,其权力大约为总统的权力的两倍半。"

Freebie Perks 盯着她有片刻,然后无所畏惧地正视着 Hogg 说:"我想我们会坚持现行的制度。"



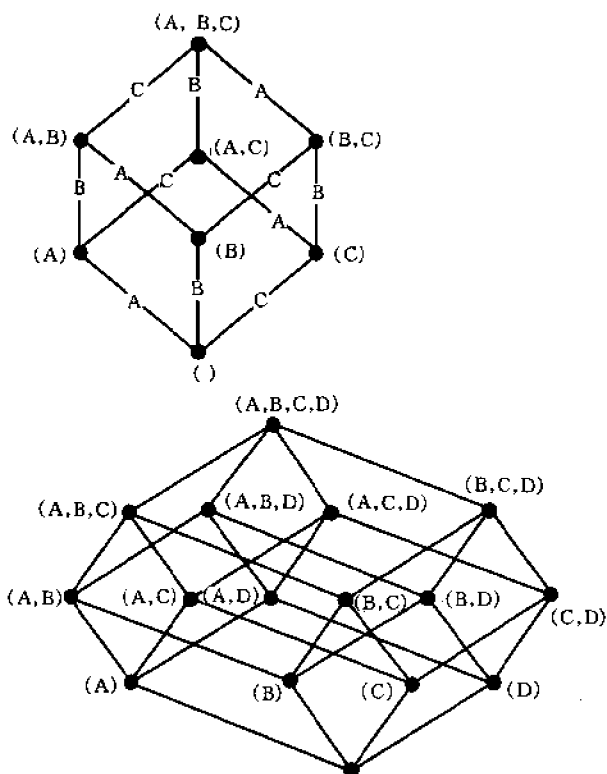


图 3　一个有三位成员的投票体制的点阵图(上)和一个有四位成员的投票体制的点阵图(下)

24. 花的发育与黄金数



蜜蜂嗡嗡地叫着，使人昏昏欲睡，阳光照射大地，向日葵在微风中摇摆。牧羊少年 Grimes 在一棵树下打鼾，而牧鹅女郎 Bumps 则在编织雏菊绞花。突然间她停了下来。

"Grimes！我刚才发现一棵雏菊只有 31 朵花瓣。通常这种雏菊都有 34 朵花瓣。"

Grimes 坐了起来，伸伸懒腰。"真的吗？还有个具体的数目，那倒是有点怪了。不过我猜想，这种花的基因必定规定——"

"我看没有什么必定。我的意思是说，基因告诉植物如何制造叶绿素，但却没有告诉植物使叶绿素成为绿色。这是化学问题，而不是遗传学问题。"

Grimes 以前已经领教过她的这段议论。"是的，的确如此。生物形态的某些特征来源于遗传性质，而另一些特征则是由生长的物理、化学及动力学特性造成的。"

"不错"，Bumps 说，"遗传特性可以产生几乎任何东西，而物理、化学及动力学特性则产生数学上的规则性。"

"我不可能说 34 是什么很引人注目的规律"，Grimes 说道。



Bumps 把花瓣从雏菊上扯下来。“我同意你的看法。但是，植物中出现的数字——不仅是花瓣的数字，而且是所有其他各种特性的数字——通常是极为特殊的。百合花有 3 朵花瓣，毛茛有 5 朵花瓣，金盏花有 13 朵，翠菊有 21 朵，大多数雏菊有 34 朵、55 朵或 89 朵花瓣。其他的数目是很少见的，主要的例外情况是这同一批数字出现翻番，或者是所谓的反常数列(3，4，7，11，18……)。”

Grimes 搔了搔他的头。“我以前曾见到过这些数。”

“是的。3，5，8，13，21，34，55，89 这些数构成了——”

“斐波那契数列的开头，”Grimes 得意地说道。“每个数都是其前面的两个数之和。你所说的‘反常数列’也具有同样的特点。”

“不错”，Bumps 回答说，“斐波那契数出现在许多不同的场合中。看看向日葵的小花是如何排列的。”

Grimes 凑近到旁边的一棵向日葵上看着。“哇！排成螺旋形的小花——34 条螺旋线沿顺时针方向弯曲着，就像车轮的轮辐，不过是曲线形的。反时针方向弯曲着的有 55 条。”

“这是两个相邻的斐波那契数”，Bumps 补充说。“确切的数字与向日葵的种属有关，不过你看到的将是 34 朵和 55 朵，或 55 朵和 89 朵，或甚至 89 朵和 144 朵。雏菊也一样。而菠萝则有 8 行鳞苞——即那些钻石状的小片——向左偏斜，13 行鳞苞向右偏斜。挪威云杉的球果有 5 行鳞苞朝一个方向，有 3 行鳞苞朝另一个方向。它们全都构成了斐波那契数对。”

“真是奇妙”，Grimes 说。

“是彻头彻尾的怪事”，Bumps 同意他的说法。“如果遗传特性可以使一朵花有随便多少朵花瓣，或使松球果有随便多少片鳞苞，那为什么会出现这么多的斐波那契数呢？”

Grimes 捻了一下他的手指。“你是说这些数字是因某种数学机制的作用而出现的？或者是物理特性，或化学特性，或——”

"动力学特性",Bumps肯定地说。

"有没有人真的解释过植物的生长是如何产生斐波那契数的?"Grimes问道。

"嗯,许多人已经提出了种种不同的答案。不过我认为,最引人注目的见解是巴黎统计物理实验室的 Stephane Douady 和 Yves Couder 提出的。他们最近运用计算机模型和实验室实验,证明了用植物生长的动力学特性可以解释这些斐波那契数——以及其他许多现象。"

"基本的设想还是那一个",Bumps继续说道。"如果观察一下一棵正在生长的植物的嫩芽顶端,那么你就可以发现植物的所有主要特征——即叶、花瓣、萼片、小花或其他诸如此类的东西——的雏型。顶端的中心是一个不含任何专门特征的圆形组织区,称为顶。围绕着顶的是一个接一个出现的细小隆起,称为器官原基。每个器官原基逐渐离开顶,最终发育成叶、花瓣之类的东西。所以,你必须说明为什么你会在器官原基中发现螺旋形和斐波那契数。"

"怎么说的呢?"Grimes问道。

"第一步是要认识到,人眼最容易看到的螺旋形——即斜列线——并不是最根本的。如果你按照器官原基出现的次序来考察器官原基,最重要的螺旋形便形成了(见彩图6)。器官原基出现得越早,它移动得就越远,所以你可以根据它们到顶的距离来确定其顺序。你将发现,器官原基沿着一条紧密盘绕的螺旋线相当稀疏地排列,这条螺旋线称为生殖螺旋。听得懂我说的吗,Grimes?"

"没问题。不过,为什么器官原基会形成螺旋线呢?"

"这个后面再说。结晶学的先驱者 Auguste Bravais 和他的弟弟 Louis 在1837年观察到一个极为重要的定量特性。他们从每个器官原基的中心向顶的中心画一条线,并测量了从顶的中心看去相邻的各器官原基所成的角。看看29号和30号器官原基之间所成的角,



或30号和31号器官原基之间所成的角。你注意到什么没有?"

Grimes斜眼看了一下。"这两个角看起来似乎相同。"

"正是如此。相邻的角几乎相等。它们共同的值称为发散角。从角度上说,器官原基是沿生殖螺旋等距排列的。好,你看看发散角有多大?"

"很大,比直角还大。"

"不错。一般说发散角接近137.5度",Bumps看起来很得意,但Grimes却不知其所以然,并照直说了。

"取斐波那契数列中两个相邻的数",Bumps开始对他解释了。

"比如说34和55?"

"是的。现在作出这两个数的比34/55,然后乘以360度。"

Grimes掏出了他通常用来掌握饲料存贮情况的袖珍计算器。"嗯,结果是222.5稍多一点。"

"量角的大小时可以从角的外面量,也可以从角的里面量。你得到的答案大于180度,所以要用360度来减去它。"

"好的",Grimes一边说一边按键,"结果是137.5度。"

"这就对了。当斐波那契数增大时,相邻的两个斐波那契数的比值趋近于0.618 034,也就是$(\sqrt{5}-1)/2$,即所谓的黄金数,用希腊字母ϕ表示。"

"我以为黄金数是$(\sqrt{5}+1)/2$",Grimes提出疑问。

"$(\sqrt{5}+1)/2$等于1.618 034。这个黄金数既等于$1+\phi$,也等于$1/\phi$。如果把相邻的两个斐波那契数的比值颠倒过来(例如55/34=1.617 6),那么不断增大的斐波那契数的比值就趋近于1.618 034。不管怎样,整个问题的关键在于'黄金角',即$360(1-\phi)$度,也就是137.507 76度。Bravais兄弟观察到,相邻的器官原基之间的角非常接近于黄金角。"

"我懂了。"

"如果你在一条紧密卷绕的螺旋线上以 137.5 度的间隔画出一连串的点,那么,由于相邻点相互配合对齐的情况,可以得到两组相互交织的螺旋线。由于斐波那契数和黄金数之间的关系,两组螺旋线中的螺旋线数目为相邻的斐波那契数。"

Grimes 盯着在草地中飞来飞去的蝴蝶有片刻时间。"这么说来全部问题就归结为解释为什么相邻的器官原基之间的夹角是黄金角了?"

"是的。其他所有结果都可以由此推出,如果假定器官原基都是围绕着顶的边缘出现——正如 Wilhelm Hofmeister 在 1868 年所提出的那样——且都沿着径向向外移动。"

"器官原基移动的速度有无关系?"Grimes 问道。

"肯定有关系。由于植物的生长方式的缘故,随着半径的增大,移动的器官原基的速度实际上不断增加——其速度与半径成正比。"

"于是 Douady 和 Couder 的这个理论就有其用武之地了?"

"是的",Bumps 说,"他们的设想是以较早的一个重要见解为依据的。如果你把植物种子模拟为一些有固定半径的圆盘,并试图把种子尽可能密集地堆集在一起,同时使发散角保持在 137.5 度这一恒定数值上,则第 n 个种子(从最新的数到最老的)必须置于一个与 n 的平方根成正比的距离上。因此,黄金角可以使种子的堆集效率达到最高。"

"请再说一遍好吗?"

"嗯,假定你做了蠢事——用了 180 度的发散角,也就是把 360 度刚好平分成两半。这样相邻的器官原基将沿两条方向相反的径向线排列。如果你采用 90 度的发散角,则将得到 4 条径向线。事实上,如果你采用 360 度的任一有理数倍数为发散角——此角可表示为 360p/q,其中 p 和 q 为整数——则将得到 q 条径向线,而这些径向线之间有很大空隙。"



Grimes一本正经地点点头。“这样种子的堆集效率就不高了。”

“正是如此。为了提高堆集效率,需要使发散角等于360度的一个无理数倍数——这个无理数无理的程度越高,堆集效率也就越高。研究数论的理论家们早就知道,最无理的数就是黄金数。”

Grimes显得困惑不解。“你说的‘最无理’是什么意思?实数要么是无理数,要么不是无理数,对吧?”

“不错。但是某些无理数要比其他无理数更无理。前面我们说过,相邻的斐波那契数的比趋近于ϕ,因此ϕ就是2/3,3/5,5/8……这个序列的极限。这些数是ϕ的有理近似值,它们越来越接近ϕ,但永远也不会等于ϕ。观察一下这些分数和ϕ的差趋近于零的速度有多快,我们便可确定ϕ的无理程度。事实上,对于ϕ,这些差值趋向于零的速度比其他任何无理数的情况下都慢。”

“这样黄金数就因这一简单的数学特性而不同于其他任何数了。”

“你说到点子上了”,Bumps说,“噢,Douady和Couder把黄金角看作是动力学特性的结果,而不是直接以高效率的堆集为理由而要求存在这种角。他们假设相继的单元(代表各器官原基)以相等的时间间隔在一个小圆圈(代表顶)边缘上的某个地方形成。然后这些单元以一定的初速度沿径向移动并互相排斥——这个条件保证了连续的运动,并保证每个新元素出现时距其紧接着的先行元素尽可能地远。”

“你的意思是说它出现于最大的间隙中?”Grimes把话说得更明白。

“Grimes,你的英语真是妙极了。可以很有把握地断定,这样一个系统将会高效率地堆集,因此你可以预期黄金角将会自动地出现。它的确出现了,不过带有一些令人感兴趣的玩意儿。”

“比如说哪些呢?”

“有两种方法可以弄清楚将会发生的情况。一种方法就是像Douady和Couder那样进行实验。但他们不是用植物进行实验，而是在一个圆盘内注满硅酮油，并把这个圆盘置于一垂直磁场中。接着他们每隔一定时间将少量磁性流体滴到圆盘中央。磁场使这些小滴极化，因此它们就相互排斥。为了使小滴沿径向移动，他们增大了圆盘边缘处的磁场强度。所出现的图形与小滴依次滴入时间隔的时间长短有关。但是这些小滴常常排列在一条螺旋线上，彼此间的发散角非常接近于137.5度。”

“黄金角！”Grimes猛然大叫。

“Douady和Couder从计算机计算中得到了类似的结果。具体地说，他们发现，根据一个复杂的摆动曲线分支图(见图的下图)，发散角与小滴之间相隔的时间长短有关。一条曲线在相邻摆动间的每一段都对应于某一对螺旋线族数。主分支的发散角接近于137.5度，沿着主分支你可以按数字顺序找到所有可能的相邻斐波那契数对。各分支间的间隔代表动力学特性经历显著变化的分歧。”

Grimes想了好一阵子。“不过也有些分支的发散角并不接近于137.5度。”

“是的。其中主要的一个对应于反常数列。这同一个模型如果适当选择时间间隔，则可产生出斐波那契规则的最常见的例外情况以及斐波那契规则本身，这既说明了这些例外情况为什么会出现，同时也清楚表明它们实际上根本不是例外情况。但当然任何人也不会认为植物学同这个模型一样完美无缺。在许多植物中，器官原基出现的速度既可以加快，也可以减慢。事实上，一个器官原基是变成叶子还是变成花瓣常常伴随着这类变化。”

“这样植物的基因或许会影响器官原基出现的时间吧？”Grimes问道。

“正是如此，但基因不一定告诉植物如何安排器官原基的间距。



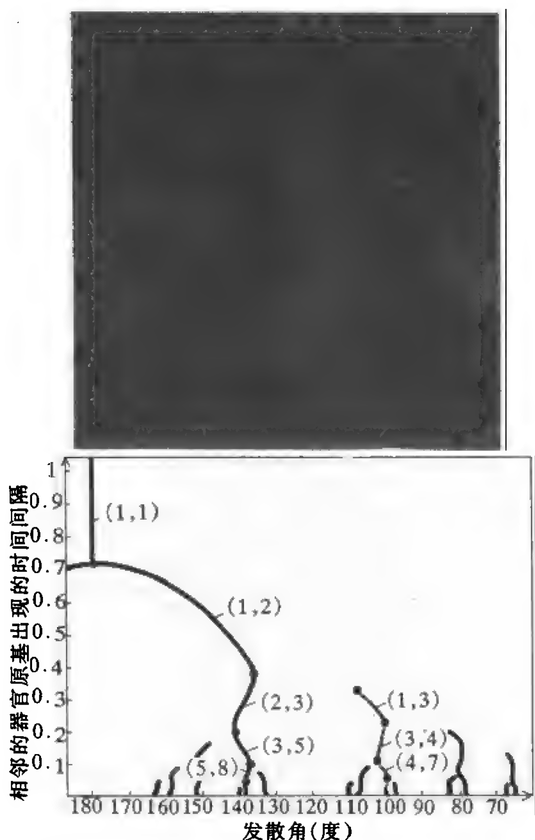


图　沿着一条紧密卷绕的螺旋线以黄金角为间隔画出若干点，便形成了本图所示的器官原基排布图(上)。这样得到的斜列线的数目与相邻的器官原基出现的时间间隔有关(下)。主曲线对应于相邻的斐波那契数对(下左)。次级曲线给出了反常数列(下右)

这项工作是由动力学完成的。这是物理学和遗传学之间的一种合作关系。”

Grimes 让一颗长得特别好的向日葵在空中摇来摆去。“你是否



认为，或许我吃了这些葵花子以后我的数学才能会有所提高?”

“试一试吧”，Bumps 说。Grimes 开始嚼葵花子。“不过——”

Grimes 停了下来。“不过什么?”

“别忘了斐波那契为什么会发明他的那个数列。或许你会变成只兔子。”

25."无中生有"的诀窍

若干年前,我收到一封很长的信,其作者声称他发明了一种能凭空制造出某种东西的方法。对此我并不感到大惊小怪。毕竟在我收到的提出这类有趣想法的来信中,总有那么些人的说法是令人难以置信的。但是,既然科学的基本精神在于要求人们有一个开放的(如果还不是完全通风的)头脑,我还是尽量设法一封不漏地阅读这类信件。

我读完上面那封长信确是一件幸事,因为作者的论断的基础恰恰是一个合理的数学结果,即所谓的巴拿-赫塔斯基悖论。巴拿赫与塔斯基是两位波兰数学家,他们在本世纪 20 年代发现了此悖论(因而这一悖论就以他们的名字命名)。该悖论揭示的是,在一定的条件下,一个理想的立体能够在分割成若干块后又组装成一个比原来大 1 倍的新的立体。

的确,这一"无中生有"方法的发明人原来是一个发表过许多值得称道的论文的专业数学家。由于种种原因(读者下面很快就会清楚),他希望避开别人的注意,并让我称呼他为 Arlo Lipof。Lipof 非



常熟悉巴拿赫-塔斯基悖论，因此他开始研究把此悖论应用到真实物质（而不是理想物质）上的可能性。

他的研究获得了相当漂亮的结果：他编写出了一个程序，它能够给出把一个立体分割成若干个奇形怪状的小块然后又拼装成一个比原来的立体大1倍的立体的精确方法，而构成这个新立体的各小块之间绝对是天衣无缝！

不消说，这个程序的意义是十分深远的。这里，我最好是援引Lipof给我的来信中的原话来解释巴拿赫-塔斯基悖论并说明该程序是如何利用这一悖论的：

“这一悖论与有名的七巧板（即剪成若干种简单几何形状的纸板）智力游戏相似。用四块这样的纸片能够组合成面积为64英寸的正方形。而还是用这四块纸片居然同样能组合成一个面积为65英寸（绝对精确的矩形）。如果你不相信，请把图上的纸片剪下来试一试。”

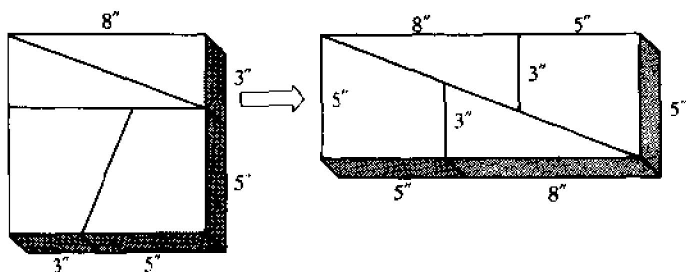


图　无中生有，得金一寸

“如果用金块来代替纸片，那么，在由正方形转换为矩形的过程中，就可以毫不费力地大发横财了。比如说，开始我们有一个边长为8英寸、厚为1英寸的方形金块。按图的左图切割，然后按右图把那些切开的小金块组合起来，就多生出1立方英寸的金子。这一立方英寸金子有4.3盎司，按时价计算大约值1 800美元！”



Lipof 接着承认上面这个例子中无中生有的现象纯粹是个错觉。但是他声称,尽管巴拿赫-塔斯基悖论也引起同样的错觉,然而这个悖论本身所依据的理论基础却是无懈可击的。至少从数学意义上讲,巴拿赫-塔斯基悖论是真实的。

巴拿赫-塔斯基悖论产生于一个已被证明了的定理,用专业术语叙述这一定理是容易理解的,如果 A 和 B 是 R^3 的任意两个有界子集,且均有一个非空内部,则 A 和 B 是可等度分解的。也可以用较通俗的语言来叙述这一定理。假定开始时有两个任意形状和大小的立体,满足如下两个条件。第一,每个立体必须是"有界的",也就是可以容纳在一个内空的有一定大小的球体中;第二,每个立体必须有一个非空内部,也就是必须能设想在立体内的某个位置存在一个球体,此球体完全被构成立体的物质所充满。

这两个条件实际上是颇为必要的,一点儿也不过分。的确几乎所有我们能想象得出的不合乎这两个条件的事物都不是我们通常称作"立体"的那类物体。例如,一条无限长的直线就违反了这两个条件。第一,它不是有界的;第二,它的内部是空的,因为它根本没有什么内部可言。想象的由点组成的向四面八方无限延伸的云也不符合这两个条件,从通常的意义上讲难以承认它是一种立体。

根据这个定理,任何两个这类有非空内部的有界立体便可"可等度分解的"。这就意味着我们可以把这两个立体分割成有限个几何上全等的小块,即其中一个立体切出的小块只要通过旋转就能与另一个立体切割出的小块相等同。因此,从理论上讲我们能够把一个立体分割成若干小块,并标之以 A_1, A_2, A_3……同时把另一个不同的立体分割成若干于小块并标之以 B_1, B_2, B_3……使得 A_1 等同于 B_1,A_2 等同于 B_2,依次类推。这便是巴拿赫-塔斯基悖论的核心。

Lipof 在来信中接着写道:"因此,取两个实心的球(其中一个比另一个大 1 倍),把它们按上述方法分割成两两全等的小块是办得到



的。让我们撇开大的那个球,只考虑小的那个球。我们想象它是用金子做的。从原则上说,这个球可以被分割成有限个小块,然后重新组合成一个比它本身大1倍的球。"

这并不是在变戏法,但我们必须清楚一点,那就在"小块"这一看似简单的词中却隐含了某些数学上的限定。首先,它们的形状不一定很简单,甚至不一定是由互相连接的部分组成的。组成同一小块的某些部分可能是彼此靠得任意地近,但实际上并没有真正相接触。其次,这些小块无法精确测量。例如,我们甚至不能设想出一个能够精确测定它们体积的方法。那么,这些小块到底是什么样子的呢?Lipof 在来信中说:"它们的样子是你以前从来不曾见过的……它们是看上去像七巧板一样的分维体。"

巴拿-赫塔斯基悖论的最普遍形式仅在三维空间或更高维的空间中成立。但是,还有一些有关的定理也能说明该悖论在低维空间内的特征。低维空间中的巴拿赫-塔斯基悖论的一个简单例子是由全部整数构成的一维"空间",因为该"空间"中的偶数子集既代表了此空间的一半,又代表了此空间的全部。之所以说偶数子集是整数集合的一半,是因为每两个整数中只有一个属于此子集。然而,通过一个简单的转换——用2除该子集中的每一项——就把偶数转换成了所有整数的集合,因此这个子集与整数集合是一样大的。

大多数人都不会认为上述事实有什么惊人之处,因为这个例子中的集合与子集合恰恰都是无穷大的。无穷大被2除仍然是无穷大。如果发现一个有限空间可以被分解为若干悖论意义上的小块,那才真正是激动人心的。但如果我们把思路限于一维空间,那么从理论上讲无论如何这是不可能的。在二维的欧几里得空间或者是"平坦"的面上同样是不可能的。然而,在某些非欧几里得二维空间中这是能实现的。

但证明巴拿赫-塔斯基悖论的最一般形式勿需用双曲线或双曲



空间。事实是,在欧几里得三维空间(它与我们所生活的空间接近)内,任何两个满足我们所能够设想的最基本的条件的实体是可以等度分解的。不幸的是,这个命题的证明是非构成性的:对于如何精确地证明两个不相等的实心球体的可等度分解性,它几乎没有为我们提供任何线索。

到此,我再次引用 Lipof 的来信中的话:"我花了许多年的时间来研究巴拿赫-塔斯基悖论和有关的结果。最使我着迷的是三维空间内的那个证明的非构成性的特点。尽管数学家们知道一个实心球在理论上能够被分割成有限个小块后再组合成一个比原来大一倍的实心球,但是,没有人知道那些小块看上去像什么样子,因为小块的分割基于集合论学者们所谓的选择公理。"

"选择公理之所以得名并不是因为数学家们偏爱它,而是因为它提出了这样一个公设,即:任何一组集合,不管有多大,都有一个从这组集合的每个集合中选择出一个元素的方法。实际上,许多数学家宁愿不用这个公理,因为它没有规定如何进行匹配。"

"因此,在我进行这方面的研究之前,人们一点儿也不知道按悖论分解得出的小块会是什么样子。我在巴拿赫-塔斯基悖论的证明中使用了机械化的方法。这一证明方法规定第一个球分割成的块可以以两种方式围绕球的中心旋转而组合成第二个(更大的)球,类似于双曲空间中的情形。旋转使得第一个球分割出来的每个小块移动到第二个球中的相应位置上。知道了构成每个小块的点和必须旋转的度数,就容易编出一个把实心球分割成各个小块的回溯程序。每当证明过程中要用到选择公理时,我只消借助于我的个人计算机中的随机数生成程序来选择球中的哪些点将是哪些集合中的元素。"

"老实说,在进行这项研究的过程中,我一点儿也没想到我会迈进了发明一种无中生有诀窍的大门。我并不傻,我知道一般情况下人们必须把数学上的理想空间与我们生存的空间严格区别开来。但

是，当我们完成了对巴拿赫-塔斯基悖论的首次模拟时，我意识到我已经获得了使任何实体变大1倍的诀窍。"

"我产生了用真实物质进行试验的想法，但最初我不敢轻举妄动。我的程序所作出的小块的尺寸是用3倍精度的数字表示的，这种精度相当于在制造小块的过程中必须把原子一分为二！而且，这时我开始怀疑自己的神经是否还正常，我清楚地感到把一个实心球实实在在地进行切分的这种想法是不现实的，我仿佛陷入了一种梦幻境界。"

"我反复对自己说，这种想法当然无论如何是行不通的。但这一点用处也没有。最后我终于一刻也不能再等待下去了，我要马上开始做这个试验。我花了我毕生积蓄中的很大一部分买了12盎司金子，并把这些金子拿去铸成了一个球。我买了一把珠宝匠用的小锯子，便开始按照程序给出的方法切分这个金球。另一个辅助程序在切分中十分有用，它把每个小块的大小和形状进行分类编目。特别是，这个程序告诉我各个小块应该装在第二个球的什么位置上。"

"这个试验从头到尾花了7个月时间。我把周末和夜晚的时间都用上了。最后我完成了切割工作，并着手把那些小块装配成直径比原来的球大1倍的另一个球。这是一件仔细而又磨人的工作，我干得头昏眼花，但还是坚持做下去了。渐渐地第二个球开始成形了，但不够光滑，那些小块不像我希望的那样密切配合，在小块之间有些极细的缝隙，我非常仔细而又煞费苦心地用小镊子把那些缝隙填起来。"

"又过了几个星期，我终于作成了第二个球，我把显示这个球的表面的主要结合处的图寄给你(见彩图7)。你的读者不会从这张图上看出多大名堂：这个球的表面和其内部小块的复杂排列相比就像小孩子的戏法一样简单。无论如何，实际的球不是像图上所画的那样圆，那样光滑，而是凹凸不平的，不规整的，可以说十分难看。但



是,在我把这个金球送到珠宝店的路上,我把装着金球的布袋捏得好紧哟!当然,对我的整个工作的最终检验是把这个球熔化,看看我现在拥有的纯金是否真的比当初我买回去的纯金重8倍。"

"第二天,金店给我送来了一根重49.58盎司的纯金条——比我期望的要少。这是由于金球中有空隙的缘故。这件事不再有什么值得怀疑的了,我已经在世界上首次把巴拿赫-塔斯基悖论付诸实际应用。我陶醉在我的发现中,一连好几天都像喝醉了酒一样飘飘然。现在我对于下一步该做什么还没有拿定主意。"

收到Lipof的第一封信后,好几月我再也没有得到他的信息了。后来,去年十一月的一天邮差给我送来了一封他从一个南美国家寄来的信:

"我相信当你得知我已经在某种程度上使由小金球制作大金球的过程实现了自动化时会感到很高兴。我用上次买金子后剩余的全部积蓄在某地的一个小镇建了个车间,雇用了几个当地人来组装金球。在一个工作间里安放了几部计算机和一些组装金球用的桌子。现在,组装金球用的小块不是切割出来的,而是由工人直接铸造成形并进行加工的。组装完一个金球后,总有金子多出来,我们又用多出来的金子再组装。这样我们一个星期能够无中生有作出大约五磅金子来。这是不是炼金术士的点金石呢?"

"很快我又得继续干下去,我想我不能再给你写信了。与你通信是相当危险的。具有像我这样大本领的人总是容易患妄想狂的病。原谅我,朋友。我要做的事情还很多……"

我再也没有收到Lipof的来信。出于好奇心,从去年十二月起我开始密切注意每天的黄金价格。在几乎3个月的时间内,黄金价格一直很低而且持续下跌。这对于那些认为巴拿赫-塔斯基悖论仅仅是个数学家们的玩物的人来说,也许就是它的实用性的最可靠的证据了。



当然，我也与其他研究巴拿赫-塔斯基悖论这个问题的数学家取得了联系。我特别感谢加利福尼亚州圣莫尼卡兰德公司的 Bruno W. Augenstein，是他建议我用双曲空间作为一个范例来讨论空间的悖论性质。

尽管 Augenstein 不赞同 Lipof 的观点，但是他承认在巴拿赫-塔斯基悖论与真实世界之间完全有可能存在一种相关关系。Augenstein 在他的一篇题为"强子物理学与超限集合论"的论文中指出了粒子物理学与三维空间中物体的悖论分解之间的一种相关关系。他在这篇论文中提出了一些能直接给出大量已知的物理结果并提示另一些原则上可以检验的物理结果的类比。例如，通过与分解定理的类比可以为夸克色标与夸克囚禁现象等找到直接的解释。这些论述如果还不能使读者中的炼金术士感兴趣的话，至少可以使他们中的物理学家感兴趣。

26. 六个难题

希望下面这些互不相关的简短问题对于大多数读者来说是新的和有趣的。第一个问题十分困难,在本文结尾发表了这个问题的解答。希望那些对于解难题饶有兴趣的读者在看解答之前先去做做这个题。

1. 无法解决的问题。问题是从大于 1 并且不大于 20 的正整数范围里选出两个数(不一定不相同)。我们只把这两个数的和告诉数学家 S,只把这两个数的乘积告诉数学家 P。

S 打电话给 P 说:“我看你没有办法能定出我的和数。”

1 个小时以后,P 回电话说:“我知道你的和数是什么数了”。

后来,S 又打电话告诉 P:“现在我也知道你的乘积是什么数了。”

这两个数是什么数呢?

为了简化这个问题,这里以这两个数中的每个数的上界为 20 给出了这个问题。这就是说,这两个数的和不可能超过 40 或它们的乘积不可能超过 400。假如你能够成功地求出它的唯一解,那么你就



会看出通过提高这个上界能够多么容易地把这个问题加以推广。奇怪的是，假如把上界提高到100，解答仍然保持不变。以色列的一个计算机程序检验了直到200万的所有数而没有找出第二个解答。也许有可能证明，即使没有任何上界的话，解也是唯一的。

2. 扑克难题。每个扑克牌手都知道，同花顺子（见彩图8中左方的一手牌）赢四条（见彩图8中右方的一手牌）。

有多少手不相同的同花顺子？在每一种花色的同花顺子牌中，可以用A打头，二点打头，或者直到10点的任何一张牌打头（A可以看成最大或最小的牌），这样一共有10种可能性。由于有4种花色，所以有4×10即40手不同的同花顺子。

有多少手不同的四条呢？只有13手。假如有13手四条和40手同花顺子，为什么同花顺子要赢四条呢？

3. 石油国的再分配。石油国酋长为他的酋长国提出下面的分配财富的方案。他把全体居民分成五个经济等级。第一等级是最贫穷的等级，第二等级是次贫穷的等级，依此类推到第五等级，即最富有的等级。他计划在两个、两个等级之间平分财富，先从第一等级和第二等级开始，然后第二等级和第三等级，第三等级和第四等级，最后第四等级和第五等级。平分就是把两个等级的全部财富重新平均分配给这两个等级中的每一个人。

酋长的大臣赞成这个计划，但是他建议平分从两个最富有的等级开始，然后按等级自上到下而不是由下到上地进行平分。

那么，最贫穷的等级希望采用哪种计划？最富有的等级希望采用哪种计划？

4. 每小时50英里。一列火车沿着一笔直轨道行驶500英里，它以正好每小时50英里的平均速度走完全程。然而，它在途中是以不同速度运行的，停车多次，甚至有时还后退。似乎有可能，在这500英里的轨道上，没有任何一个50英里的路段，是火车正好用1



个小时通过的。

问题是证明,情况并非如此。

5. 隔子跳"啊哈!"。在一张纸上画一个五行六列的位点阵。如图1所示。画一条直线把这个位点阵二等分为两个三角形的位点阵,每一个各有15个位点。在直线上方的15个位点(图中表示成涂黑的)上放15个分币或任何其他类的小东西。

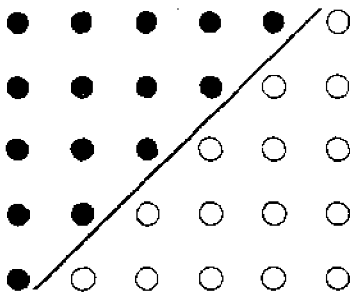


图1 一个"啊哈!"问题

问题是把分币全都从直线上方的位点移到直线下方的位点上。每次移动是把一个子跳过相邻的子到对面紧接着相邻子后边的一个没有子的空位点上。可以向左、向右、向上、向下跳,但是不能斜跳。举例说,最上面一行第四位点上的分币走第一步时可以跳到最上面一个未涂黑的位点上,也可以向下跳到从此分币所在的一列的最上面算起第三个位点上。所有跳法都跟西洋跳棋跳法一样,不同之点只是局限于水平跳及垂直跳,并且跳过的子不拿掉。

我们不涉及以最少步数把这些分币搬到未涂黑的位点上这件事,而只涉及究竟是否能够搬这件事。有三个疑问:

A. 搬的任务是否能够完成?

B. 如果从一个涂黑的位点上搬走1个分币,那么余下的14个

分币是否能跳到未涂黑的位点上?

C. 如果从二个涂黑的位点上搬走两个分币,那么余下的 13 个分币是否能跳到未涂黑的位点上?

这个新问题特别有趣味,因为 10 岁的小孩能够"啊哈!"一声顿然醒悟而迅速地解答出所有上述三个疑问来。

6. 环面的悖论。两位拓扑学家吃午饭时讨论两个环连的曲面,如图 3 的左方所示,其中一个是画在一张纸餐巾上。你不要把这些东西想成是实心的,就像绳索或者实心橡皮圈那样。它们都是环面,一个是亏格为 1 的曲面(一个孔洞),另一个是亏格为 2 的曲面(二个孔洞)。

我们按照"橡皮条几何学"的方式进行思考,假设图中的曲面可以任意地拉伸或收缩,条件是不要拉破或把分开的部分粘在一起。那么,是否能把二个孔洞的环面变形得使其中一个孔洞成为不环连(如图 2 右方所示)的呢?

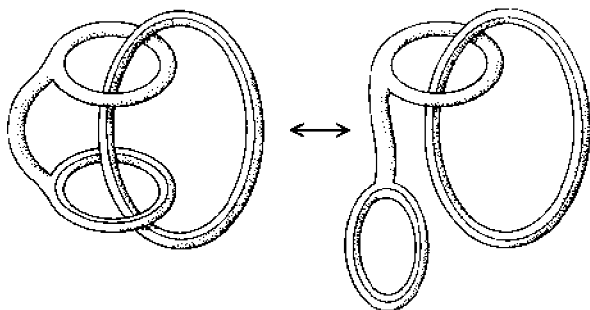


图 2 Herberl Taylor 的环面的悖论

拓扑学家 X 提供了下面这个不可能性的证明。在每一个环面上画一个圈,如图中红线所示。这两个圈在图左方环连,在图右方不环连。

于是 X 说:"你一定会同意,嵌入在三维空间中的两个环连的圈



不可能通过连续变形而变成不环连的。因此可以推定,这种变换是不可能的。”

Y说:“但是,根本不能推定。”

那么,谁是对的?

问题1(“无法解决的”问题)的解答如下:这两个数是4和13,不难记住,因为一副扑克牌有4种花色,每种花色有13张牌。因此S的和数是17,P的乘积是52。下面就是一个解答的步骤。

S说:“我看你没有办法能定出我的和数”之后,P很快就认识到这个和数不可能是两个素数之和。为了了解为什么,我们假设和是14。S就会进行如下推理:“或许这两个数是素数3和11。由于它们的乘积33只有一对因子3和11(因子1和33可置之不理,因为每个数都要大于1),P马上就会知道我的和是3加11,即14。”因此,当S说,P不可能知道他的和时,也就告诉了P,这个和不是两个素数之和。

数论中有一个著名猜想叫哥德巴赫猜想,指出任何偶数都是两个素数之和。这一点还没有得到一般的证明,但是对于直到1亿的所有偶数,已经得到证实。因此,我们可以有把握地排除从4一直到40这一给定范围内的所有偶和数。我们可以排除得更多些。由于2是素数,我们还可以排除所有那些等于一素数加2的奇和数。在完成这些排除之后,只剩下7个可能的和数:11、17、23、27、29、35、37。这些是给定范围内仅有的一些等于一复合数加2之和的奇数。

此和数可能是11吗?不是。假如P的乘积是24,他马上就会断定,两个数是3和8,因为只有24的这两个因子才能构成和数11,即上述7个可能数之一。另外,假如P的乘积是28,他也会知道和数是11,因为只有因子4和7加起来才能等于和数11。那么,S就不能做出他最后的断言说他知道P的乘积,因为他无法在24和28这两个乘积中作出抉择。结果,11作为一个可能的和数而被排除。



此和数可能是23吗？不是。S不能够在4+19和16+7(其和数均为23)这两者中作出抉择，他就不会知道乘积是76还是112。用同样的方法可以排除27，因为4+23=8+19=16+11=27；也可以排除29，因为16+13=4+25=29；也可以排除35，因为4+31=16+19=35；也可以排除37，因为8+29=32+5=37。现在只剩下个可能的和数：17。

有七对可能的数加起来等于17，列举如下：

2+15。乘积30可能被P看作是5×6的积，并且由于5和6加起来等于11这一可能的和数，所以P就不能够判定17和11这两个和数哪一个是正确的。

3+14。乘积42可能被P看作是2×21。2和21加起来等于另一个可能的和数23，从而造成了和上面一样的不确定性。

5+12。乘积60可能是3×20，它具有可能的和数23。

6+11。乘积66可能是2×33，它具有可能的和数35。

7+10。乘积70可能是2×35，它具有可能的和数37。

8+9。乘积72可能是3×24，它具有可能的和数27。

现在只剩下一对数4+13。乘积52只有另外一对因子2×26，它们加起来为28，不是一个可能的和数。因此，只有当两个数是4和13时，P才能够肯定知道S的和数是17。由P知道此和数这一事实，S就可以推出P的乘积是4×13=52。任何另一对数都具有不确定性，不能使S与P都知道这两个数。由于不确定性，随着数变大而增加，所以猜想甚至当把上界去掉时也不会有其他的解答看来是合理的。

对那些可能想证明当把上界提高到100时该结果亦然成立的读者，需要补充一句，更大的可能的和数是41、47、51、53。假如把这个上界提得更高，要排除其他的解似乎就需要计算机程序。

27. 西洋跳棋

其乐无穷

我们对于西洋跳棋的起源一无所知,尽管大多数棋艺史家现在认为它于 12 世纪的某个时候起源于法国南部。现在国际象棋的规则在整个西方世界都已标准化了,但是西洋跳棋的规则可不是这样。在讲英语的国家之外有几十种按当地规则的下法。在欧洲和前苏联最流行的下法称之为波兰跳棋(波兰除外,在波兰称为法国跳棋),它是在 10 行 10 列的棋盘上下的,开始下时双方各有 20 个棋子。这是标准的法国式下法。在加拿大法语区,棋盘要更大一些:12 行 12 列,每方有 30 个棋子。全世界下西洋跳棋的规则差别很大。

西洋跳棋比国际象棋简单,这个事实产生一些后果。一个后果是:一位西洋跳棋大师犯一个错误就不像国际象棋大师那样招致一败涂地。对于跳棋迷来说,这就是跳棋的一个有很大吸引力的地方。

西洋跳棋规则简单的另一个后果是:到 1900 年左右,跳棋的开局已经分析得如此充分,以致大多数比赛以平局告终。为了使跳棋玩法的花样更多,英国(在 1900 年左右)引进这样一种做法:把黑棋走的第一步和白棋跟着走的一步这两步走法的各种组合方式分别记



在卡片上。在每一次比赛之前,随意抽取一张卡片,比赛必须按照卡片上规定的这两步走法来开局。因为每一方可以在七种走法中选择,因此共有49对可能的组合方式。其中两对(9～14、21～17及10～14、21～17)被排除掉,因为它们要使一个白子被吃掉。后来又发现另外两对组合方式(11～16、23～19及12～16、23～19)使黑子占太大的优势,因此也必须排除掉,这样就剩下45张卡片。

标准的西洋跳棋是根据对图1所示的方格加以编号来表示的。为了图示清楚起见,习惯上把跳棋棋盘中方格的颜色掉换一下,然后在白方格中表示棋子而不在黑方格中表示棋子。真正下棋则总是在黑方格中走,每位棋手的右下方有一"双角"。习惯上把棋手称为黑方和白方,即使棋子是红色和白色的也这样叫。现在比赛时用绿色-米色方格棋盘,而黑色-红色方格棋盘被认为是玩具店的次品。总是

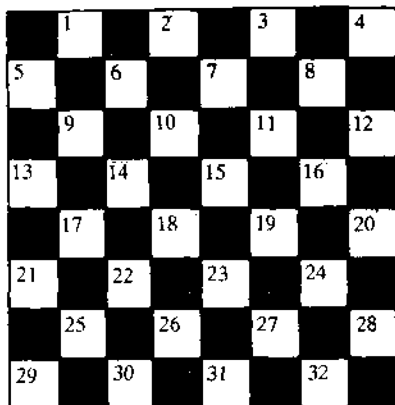


图1　西洋跳棋表示法中的方格编号

黑方先走,比赛记录总是由占有编号较小的方格的黑方开始。如果你要玩本专栏内任何游戏的话,那你最好把你的棋盘黑方格如图1所示标上号码。



几十年过去之后，专家们很快就通晓遵照两步开局走法的各种变化花样，结果采用了“安全”比赛法，从而又再次出现一个又一个的平局局面。于是在30年代中期在美国就用“三步限制”来代替英国的“两步限制”，目前在美国和英国的大多数跳棋比赛中都采用这种办法。一共有142张卡片，每张表示一种不同的头三步走法。因为这些头三步走法中，有许多走法使某一方获得优势(通常是走第二步的棋手)，所以每抽一次卡片都比赛两盘，使得每位棋手各在一盘中先走。

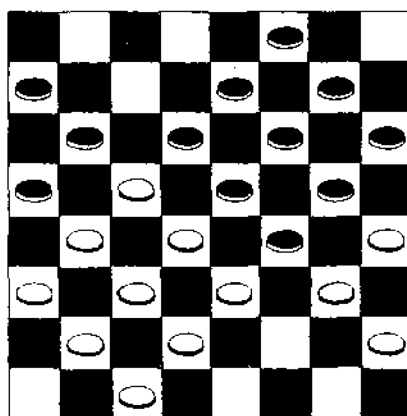
按照没有开局限制的走法，即所谓随便你走的玩法，专家们比赛就只能下成平局。即使采用三步限制，所有比赛中仍有80%左右以平局告终。如果一位专家的确赢了棋，通常是因为输家犯了个大错，或赢家设法使他所发现的“绝招”加以保密(有时保上好多年)。正如在国际象棋中一样，绝招是对标准“书本赛法”的一种改进，它使对手大吃一惊。按照惯例，每位棋手走一步之前只容许有5分钟的思考时间，在只有一种方法吃子的情况下只容许有1分钟的思考时间。近年来，这种办法被使用象棋钟所取代，每位棋手在1小时之内规定走30步。如果某人突然拿出一个新的绝招，他的受害者根本就没有足够的时间来分析它。

西洋跳棋规则简单的第三个后果是：中等水平的跳棋手要赢最好的下跳棋的计算机程序，比起中等水平的象棋手要赢最好的下象棋的计算机程序，要困难得多。杜克大学的两名研究生设计出一个强有力的非学习型下跳棋程序，一般称为杜克程序。西洋跳棋棋手分成三级：初级、高级和大师。杜克程序的支持者相信它一开始可在大师级水平比赛。但是，一位特级大师同这个程序比赛一会之后，就能觉察出它的弱点并开始利用这些弱点。它最大的弱点就是它比赛时没有通盘计划。甚至在开局时它也不遵照书本上的标准走法，通常把它的棋子散布到整个棋盘上形成一个特级大师看来是相当笨的



棋局。它的威力就在于它能够以惊人的速度分析所有可能的走法而达到比人类对手更为深刻的程度，而且在这种深度范围内，它总也不会犯错误。

在国际象棋比赛中，不难证明“将死傻瓜”是所有可能的象棋比赛中最短的比赛。所谓“将死傻瓜”就是第二个棋手在走第二步时，就把对方将死。令人惊讶的是迄今还不知道最短的跳棋比赛。在两年之前，曾认为图2所示的24步堵死结局是最短的跳棋比赛，图中棋盘显示最后的棋局。有许多种24步的走法序列能够产生这种棋局，但是，这种棋局却被认为是很独特的。在给出的走法序列中，白方是按照黑方上一步的走法成对称地(对于棋盘中心)对着走。这里给出的走法由两步的爱丁堡开局开始。因为新手最喜欢用的第一步走法9～13，对黑子来说被认为是最坏的开局；所以对称走法的比赛更经常以10～15，23～18开局，这种开局被称为凯尔索十字。

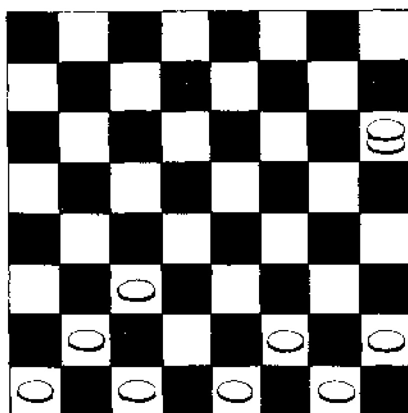


黑方	白方
1.9～13	24～20
2.12～16	21～17
3.10～15	23～18
4.15～19	18～14
5.8～12	25～21
6.4～8	29～25
7.6～10	27～23
8.10～15	23～18
9.2～6	31～27
10.6～9	27～24
11.1～6	32～27
12.6～10	27～23

图2　最短的无吃跳棋比赛



24 步堵死结局的确(能够被证明)是最短的天吃比赛。但是 Alan Malcolm Beckerson 在 1978 年发现一种走法,其中白方在走第 10 步(双方共已走 20 步)时取胜,这时白方已把所有黑子都吃掉了!这是现在已知的最短跳棋比赛,虽然还没有人证明不可能有更短的比赛。Beckerson 还发现另一种吃掉所有黑子的 20 步走法以及在吃掉一些黑子之后最终以堵死结局的 20 步走法。图 3 中这种走法,图上棋盘表示最终棋局,其两步开局称为纽卡斯特开局。



黑方	白方
1.11～16	21～17
2.10～14	17×10
3.6×15	23～18
4.2～6	18×2(成王)
5.9～14	2×18
6.3～7	24～20
7.1～6	20×2(成王)
8.12～16	2×9
9.5×23	26×3(成王)
10.4～8	3×12

图 3 已知的最短跳棋比赛

有许多其他最少步数的跳棋难题还远未解决。至少用多少合乎规则的步数能够使一盘棋产生 24 个王?现在人所共知的解是用 90 步。至少用多少步可以使黑子和白子互换它们的初始位置?每一方只在棋盘上至少走 60 步才能占上对方的初始方格,因此,$2\times 60=120$ 步是绝对下界。19 世纪末有人给出一个 172 步的解答。最后双方各有六个王。似乎 172 步这个步数还能够大大减少。

对于这个互换初始位置的难题,在更小的棋盘上进行尝试也是非常有趣的。3 行 3 列的棋盘没有什么有意思的结果,但 4 行 4 列

的棋盘就提供一个有趣的难题。棋局的开始如图 4 所示，问题是用最少的合乎规则的步数把双方互换一下。最后，这四个子都必定成为王。顺便插一句，在这个小棋盘上，最短的比赛需要五步。如果双方下棋是为了赢棋，都使用他们的最佳战略，那么比赛结果是平局。

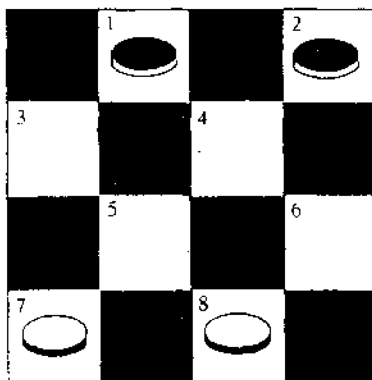


图 4　小棋盘跳棋的难题

正如象棋的情况一样，通过改变棋盘大小、初始棋局、下棋规则等等，已提出了无穷多种下跳棋的玩法。有的在三角形或六角形棋盘上下，有的在三维棋盘上下，有的在跳棋子中还混进象棋子，有的容许三个或四个棋手同时参加比赛。正如人们可以想象到的，有的跳棋玩法非常相似可以称为跳棋的一个变种，有的玩法差异很大以致最好看成完全是另外一种游戏，在这两者之间很难划清一条界限。例如，所谓土耳其跳棋，除了它在 8 行 8 列的棋盘上用两种颜色的棋子下之外，几乎和西洋跳棋没有任何相似之处。改变标准跳棋的一种简单办法是开局时的棋子按照图 5 的方式来摆。所有跳棋规则都仍然遵守。一开局就很快地导致在正统跳棋比赛中永远碰不到的棋局。

在英国和美国最流行的跳棋变种是“输子”。它与标准跳棋不同



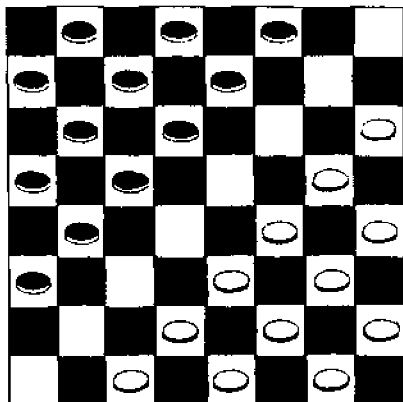


图5　对角跳棋

之点只是下棋的目标是首先把自己的棋子都输光。有一个奇妙的、输子的"勾引赌博",可能是英国的跳棋骗子设计出来的。白方开局时有12个棋子处于通常的开局位置上。黑方只在方格7处有一个王。假如黑方输掉他的王,那他就赢了。假如白方的12个棋子都被吃掉,那他就赢了。Dunne指出了,白方如何走就始终可以赢棋。他还给出三个类似的赌法,其中黑方在开局时,只在方格1、或方格4、或方格5处有一个没有加冕成王的棋子。

在几百个跳棋骗子赌博中,最好的一个是开局位置如图6所示者。现在轮到黑方走。白方赌的是黑方不能使他先走的棋子加冕成王。显然黑方不能动方格21中的棋子,因为那就会使这个棋子马上被吃掉,因此问题是黑方能否走动方格19中的棋子,并把它推进到使它成王的一行。你越研究这个棋局,似乎黑方能够轻易赌赢就越明显。然而,白方却赌赢了。这是一个同朋友玩的很有趣的打赌。

最后一个问题是:人们普遍相信,二个王总能赢一个王。但这并



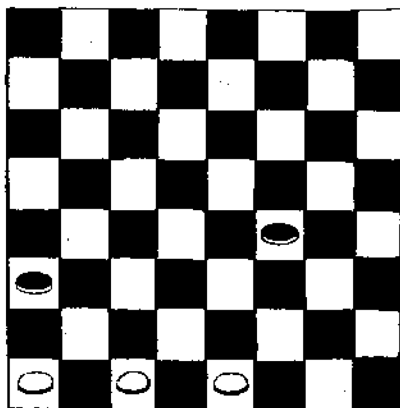


图 6　一种跳棋骗子的打赌

非总是如此。想想你是否能够在棋盘上摆上两个白子王和一个黑子王使得甚至在轮到白方走时,黑方也能迫成平局。

28. 复原洗牌法

大多数扑克游戏开局时都需要有个人洗牌。洗牌的目的自然是把各张牌出现的次序打乱，使其变得无规则。然而，如果洗牌洗得太过分完善，那么所得的结果就远不是随机的。试考虑一下人们熟悉的交叉洗牌法(riffle shuffle)，这种洗牌法就是把牌分成两叠，然后让两叠牌互相交叉起来。

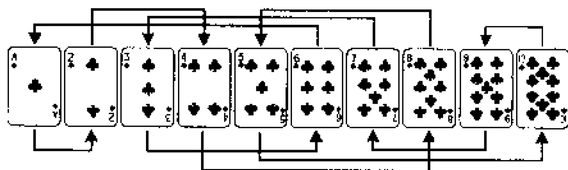
为了论述方便起见，假定一叠扑克牌有 10 张牌，所有牌都是同一花色，A 放在这叠牌的最上面，然后依点子由小到大的顺序排列下去，10 点牌在最下面。如果在洗牌后，原先这叠牌的上面一半的第一张牌仍然在洗过后的牌的最上面，则各张牌的次序就变成了 A,6,2,7,3,8,4,9,5,10。如果原先这叠牌的下面一半的第一张牌在洗牌后跑到上面来了，则这些牌的次序就变成 6,A,7,2,8,3,9,4,10,5。第一种方法称为"外洗"(out-shuffle)，第二种方法称为"内洗"(in-shuffle)。

关于交叉洗牌法的有案可稽的最早记载可追溯到 1726 年，是在一本名为《现代游戏技巧与奥秘大全》(Whole Art and Mystery of

Modem Gaming，作者不详）的书中提到的。1843 年，J. H. Green 在《赌博技巧与奥秘大曝光》（An Exposure of the Arts and Mystries of Gambling）一书中向美国人介绍了交叉洗牌法。内布拉斯加州的牧场主 Fred Black 是一位早期的交叉洗牌者，他在马背上练就了交叉洗牌的技术，并研究出标准的 52 张牌的反复外洗的许多数学道理。1957 年，英国魔术师 Alex Elmsley 公布了关于任意张牌的扑克牌的许多主要定理。

一次外洗可以看作是对少两张牌的一叠牌进行的内洗。如果我们让 A 牌留在有 10 张牌的一叠牌的最上面，10 点牌留在最下面，并对第二至第九张牌进行内洗，那么得到的结果就是 A，6，2，7，3，8，4，9，5，10——同对所有 10 张牌进行一次外洗得到的结果是一样的。这一关系使我们可以只考虑这两种洗法中的一种。

10 张牌



8 张牌

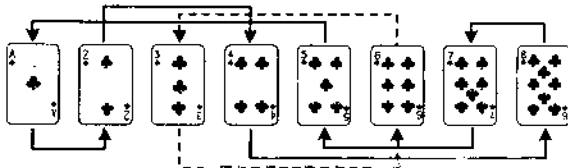


图 1　对有 10 张牌的一叠牌进行洗牌，就使它们通过有 10 步的循环过程反复变换其位置。对 8 张牌的一叠牌进行洗牌则有两种循环方式

假如我们对有 10 张的那叠牌反复地进行内洗，会发生什么情况



呢？这叠牌的次序会不会变得越来越乱了呢？初看起来洗牌似乎是使这叠牌的顺序变得没有规律——经过三次洗牌后，10张牌的顺序变成了这样：7，3，10，6，2，9，5，A，8，4。但是经过5次洗牌后，整叠牌的次序变得恰好和它的最初顺序完全相反了！显然，再洗5次牌将会使最初的顺序"复原"。我们可以得出这样的结论：反复地对10张牌进行内洗，就使这些牌在10种不同的次序间不停地循环。而这只是10张牌的3 628 800种不同排列方式中的极小部分。

如果你尝试对有偶数张牌的任何一叠牌进行这种反复的内洗，那么就会发现，在重复进行了足够多次洗牌后，这叠牌总是回到它原先的顺序上。为什么这种反复循环不可避免呢？图1说明了对有10张的一叠牌进行内洗时，每张牌的位置是如何变化的。例如，A移到2点的位置上，2点移到4点的位置上等等。跟踪箭头的移动，我们可以看到这些牌按以下次序互相取代：A→2→4→8→5→10→9→7→3→6→A。每洗一次牌，这10张牌就沿着上述循环过程向前推进一步。由于这一循环总共有10步，因此，经过10次洗牌后，每张牌都回到其起始位置上了。

这叠牌的非典型的特征在于它只有这样一种循环方式。更典型的情况是有8张牌的一叠牌，它也在图1中示出。对于这样一叠牌，有两种循环过程：A→2→4→8→7→5→A，以及3→6→3。第一个循环每洗6次牌重复一次，第二个循环每洗两次牌重复一次。当第一个循环首次重复时——在洗了6次牌之后——第二个循环已经重复3次了。

不论有多少张牌，它们在一叠牌中的位置变动都可以分为一系列的这种循环。这是为什么呢？试取随便一张牌，跟踪它的位置变化情况。由于这叠牌只有有限张，最终那张牌必定回到先前它已在过的一个位置。从这一步开始，它将重复先前的各次位置变动。但是我们是否能够肯定，当这张牌第一次到达先前已到过的一个位置



上时,它所到达的是其最初的位置吗?这个问题的答案是肯定的,因为任何洗牌都是可逆的。为了解保我们能够回溯到出发点,第一个被重复的位置必须是最初位置。基于类似的理由,任何牌都不能从一个循环跳到另一个循环上。

一旦我们知道了有哪些循环,就可以通过一种简单的方法发现需要经过多少次洗牌才能恢复这叠牌最初的次序。每个循环的长度由它的步数决定:如果某一循环有 n 步,那在这一循环中的每张牌经过 n 次洗牌后就回到其最初位置。如果一叠牌有不止一个循环,那么我们必须确实所有各循环的长度的最小公倍数。例如,假定某叠牌有两个循环,第 1 个经过 10 步后重复,第 2 个经过 14 步后重复。那么在第一个循环中的各张牌在洗了 10 次、20 次、30 次、40 次、50 次、60 次、70 次等等后将回到其最初位置,而在第二个循环中的各张牌在洗了 14、28、42、56、70 次等等后将回到其最初位置,这两组数中第一个共有的数为 70,它也就是 10 和 14 的最小公倍数。到洗第 70 次牌时,所有各张牌都回到其最初位置上了。

无论一叠牌有多厚,交叉洗牌法总是会使这叠牌最初的次序重现。但是所需要的洗牌次数以及这叠牌的张数之间是否存在某种关系呢?有 10 张的一叠牌在洗了 10 次后就开始重复,其洗牌次数等于牌的张数,但这一事实并不具有普遍性。事实上,当一叠牌有 4 张、6 张、8 张、10 张、12 张、14 张、16 张、18 张、20 张、22 张和 24 张牌时,它分别需要洗 4 次、3 次、6 次、10 次、12 次、4 次、8 次、18 次、6 次、11 次和 20 次牌才回到其初始次序。这中间存在着一种规律,但你必须是数论专家才能发现它!

以 10 张牌为例。图 2 中的表示出了 2 的各次幂以及它们被 11(也就是牌的张数再加 1)除以后所得的结果。对于 2 的 10 次方,这一余数为 1,而使有 10 张的一叠牌的顺序复原所需的内洗次数为 10。对于 8 张的一叠牌,我们看看 2 的各次幂被 9 除以后所得的余



数。对于 2 的 6 次方,这一余数为 1,因而使有 8 张的一叠牌的顺序复原所需的内洗次数为 6。

10 张牌的情形

指数(n)	1	2	3	4	5	6	7	8	9	10
2 的乘方(2^n)	2	4	8	16	32	64	128	256	512	1 024
2^n(10+1)的余数	2	4	8	5	10	9	7	3	6	1

8 张牌的情形

指数(n)	1	2	3	4	5	6
2 的乘方(2^n)	2	4	8	16	32	64
2^n(8+1)的余数	2	4	8	7	5	1

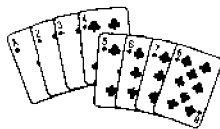


图 2　把 2 的各次幂分别除以一个适当的数直到其余数等于 1,就可以得到使一叠扑克牌恢复其最初的顺序所需要的内洗次数

这一规则普遍成立。使余数为 1 所需要的 2 的乘方的次数始终等于使一叠牌的次序复原所需要的内洗次数。此数永远小于或等于这叠牌中的牌的张数。由于外洗与一叠牌去掉两张后的内洗实际上是相同的,因此对于外洗存在着类似的规则,但是现在要用 2 的乘方除以牌的张数减 1。

对于标准的一叠牌(52 张),需要 52 次内洗才能恢复其最初的次序,但只要 8 次外洗就可以恢复其初始次序。检验这一结果是很麻烦的;即使是对于专门的魔术师,完全正确地搞一次交叉洗牌(它要求两叠牌恰好互相交错地插入)也是相当困难的,至于要一连洗几次牌而不出一点问题则几乎是不可能的事情。反过来搞比较容易一些,也就是把一叠牌轮流发给两个人,然后再把他们手中的牌叠在一起。与内洗相反的过程称为内分(in-sort),与外洗相反的过程则称为外分(out-sort)。无论你是洗牌还是分牌,使一叠牌恢复最初的顺序所需要的步数是相同的。

29. 变佳作为梦呓

如果给计算机程序输入弄得非常混乱的信息让它进行处理，则几乎任何一种计算机程序都可以产生出一堆毫无意义的结果来。有一句老掉了牙的谚语"进去的是废物，出来的就是垃圾"，说的就是这个意思。这道理现在已是如此确凿，以致没有什么人会有意于再去作一番论证。然而，只需稍加思索并略费力气，便可作出这样一个程序，它输入的是不朽的文学巨著，而输出的却是十足的胡言乱语。进去的是《麦克佩斯》(莎士比亚名剧)的最后一幕，出来的却是一位白痴的胡诌，从头到尾全是毫无任何意义的大吵大闹。

变文学作品为胡言乱语的过程可通过两步来完成。首先由计算机程序"读"一段文本，从中得出并记录下某些统计性质，这些统计性质确定了源文体中任一给定字母跟在另一字母(或另一串字母)之后的概率。第二步就是根据所记下的概率来随机地选择字母，由此产生出一段新文体。其所得结果便是一大堆字符，它们重现了源文本的统计性质。然而，如果说它有任何意义的话，那也仅仅不过是碰巧而已。



耶鲁大学的威廉·拉尔夫·贝内特详细研究了这种产生随机文本的过程。

他指出,现在已知的最早提到语言的随机生成的书是17世纪90年代坎特伯雷大主教约翰·提洛森所写的《箴言和讲道集》。他在该书中举例说明神的创造性时写道:"如果把一个口袋里的一堆字母搅乱,然后把它们随便扔到地上,那么需要搞多少次才能碰巧有一次这些字母落到地上时组成一首完美的诗,或者甚至是组成一篇优美的演说辞?一本小小的书能够如宇宙这样一本内容无比丰富的大书那样轻而易举地靠碰运气作出来吗?"

当代人们对随机语言这一问题的考虑,发端于阿瑟·埃丁顿爵士于1927年所说的话:"如果一大群猴子在若干台打字机上胡敲乱打,则它们也可能打出大英博物馆中所收藏的全部书来。"埃丁顿的本意是要强调这样一种结果的不可能性;他是把它当作一个例子,说明有的事件在原则上可能发生,但实际上却永远不会发生。虽然如此,自埃丁顿以来,在猴子的胡敲乱击中找出创造性的这样一种可能性本身就已具有了一种文学上的生命力。贝内特提到了Russell Maloney及小Kurt Vonnegut的著作,以及Bob Newhart所写的一幕夜总会的剧。

埃丁顿所设想的过程可以用一个程序来加以模拟,我把该程序称之为零级文本生成程序。首先取定一张字母表(即字符集),它决定了在猴子所敲击的打字机上究竟要装哪些键。在某些较高级的模拟过程中,使键数保持最小是非常重要的。为了前后一致起见,在零级文本生成程序中看来最好也取相同的字符集。所以我采纳了贝内特的建议,选择了由28个符号组成的一个字符集,其中包括26个大写字母,1个空白符(计算机把这个符号与其他任何符号完全一样地加以处理)以及1个撇号(即在多数书面英语中这符号较之用得最少的三个或四个字母还出现得多些)。

在任何时刻，埃丁顿所虚构的那些不带任何先入之见的猴子都有同等概率敲击打字机的任一个键。这个行为可用一个简单方法加以模拟。给字符集中的每个符号赋与从 0 到 27 之间的一个数。如要产生一个字符，就随机地选择从 0 到 27 之间的任一数，并打印出与该数相应的字符。图 1 示出了用这种方法所产生出的一小段文本。它与书面英语或任何其他一种人类语言都完全不同。它的“单词”往往是异乎寻常地长(平均有 27 个符号)，并且辅音太多。之所以如此，自然是因为在真实的英语文章中，字母出现的频率远不是均匀的。单是空白符，通常便要占字符总数的 1/5 左右。而 J、Q、X 与 Z 四个字母合起来还占不到 1%。但在零级模拟过程中，所有符号却有相同的出现频率，即全为 1/28。

PWGMMLTHIDVGRHPEDFCXFEKFNOPYPQSXZRUXG'YS'AEEU FEDEGLQYFUWPO'IKI
QTONIXJKZEUKDXWKKJREHYHPKWUJHLEJNBPLQ AIEOQXUBJYYVIFFDPQGIGZNTI
RQXPDJ NQESPQMCRSNGMKQEZICZV'GSWALK ZZEYIBBOTDCRSMK'VI MRCZXUBI
SNEQ'VQQHFQUCBJXZRVVNIBHFJEFTCFJPWFOIYHOMPNFSFWKNCMVLOJ.BX
QV KIZTLNRWGGTZFPZPQQOGVJCPAYRDQJRMYSWCGABRXLERCYYRHQCHTOQ'UT
FMRITFTIZUIWTSTXWQGOCAFXJOZYKSTV'BYOBEUFIRQWQ VOUVQJPRKJWBKPLQZCB

图 1　根据有 28 个符号的字母表作出的零级随机文本

Bob Newhart 的滑稽剧描绘的是一群检查人员所处的困境，因为他们必须阅读猴子们所打出的那些东西。一连好多个小时他们所看到的全是一堆毫无意义的胡言乱语，而后终于碰到了这样一段话：“To be or not to be, that is the gesorenplatz……”。事实上，即使要得到这个结果也是完全不可能的；这段话的前 9 个字是哈姆雷特的独白，平均要打 2×10^{46} 个字符，这 9 个字才有机会出现一次。我从计算机一次打出的 50 000 个字符中，才找出了一个“TO”和一个“NOT”，而且它们还相隔好多行(我并没有读这 50 000 个字符，而是用一个模式配对程序进行寻找)。

如要提高这些猴子们的文学才华，则第一步就是要对选择某一



特定字母的概率加以调整，以反映这些字母在书面英语中出现的真实频率。实际上，这个方案就是要作出一台打字机，其上有(比如说)2 500个空白符键，850个E键，700个T键等等。字母出现频率可以是从统计许多英语文章后所得出的平均值；但如果根据一篇特定的源文本来得出这些频率，则是更为方便，也更令人感兴趣。一个程序如是按照这样一种频率分布来选择字符，就称为一级文本生成程序。

字母出现频率值可以用一个有28个元素的一维阵列来表示。这个阵列就是计算机存贮器中的一个存贮单元块，存贮单元块的安排使得任一元素均可由一个下标(大小在0与27之间)来加以规定。为了填满这阵列，可以对文本中每一字母的出现次数进行计数，并通过人工输入所得的值。但更好的方法是让程序自己进行计数，虽然有时这意味着须事先把文本作成机器可读的形式。计数程序开始时把阵列中全部元素置零，然后一次一个字母地检查文本，一个字母一旦出现一次，对应的阵列元素值便增加1。

某一字符出现的概率规定为与该字符在阵列中的元素成正比，这样就可以产生出一级随机文本。有一种方法如下所述。首先产生出一个随机数，其值介于0与所有阵列元素的和之间(该和也就是源文本中的字符总数)。然后从这个随机数中减去第一个阵列元素，这元素也许记录的是字母A的出现次数。如果减得的结果等于或小于零，就打印出一个A；否则就从第一次比较后所剩的值中再减去下一个元素(代表字母B)。这减法要依次连续进行到得出零或负数为止，而相应的字符就被打印出来。注意，由于随机数不能超过阵列元素之和，故这种方法最终必定会得出一个选择。

图2示出了一段一级随机文本。这段文本是基于对詹姆斯·乔伊斯所写的《尤利西斯》的最后一章(该章称为"Ithaca")中的一段话，即莫莉·布卢姆的独白进行统计得出的频率阵列产生的。我选择这

段话是有根据的。由于源文本没有标点符号，因而随机文本中不加标点也就没有什么关系了。

一级

HUD T ALONIT NTA SN TVIOET ELERFOAD PE TRLTWTL N CABEG TYLUEMU TIGT
BH OFDRRIC O STU HOOOTC YATNDL UYA HWAE SS NLSDB OTRORT DEERARFT
D LBFF HHARE MW OSPE OFOIT SEOUN GTUMG H N GHKOY T EADS A SD E TNNE
PEHAGIADIHNATO AATSAGI ED INNE ABRA TAAM GT E TWNO HEWIIGUTNCM GA SFHHY
HREBH RARE OOSY LFE OC EGGTA WIFRTYE EJS DA ETO WF EIT ERNETEBTSTTELO
NTAAN O YEETWNSONRNHN TYHVN NLUESETTHLGEAKPNNMTIA TSM REEANTVONC POE
RUTP EOIT L IEETGTWHSW H KHHER W OLIOEWOEPT D AEYBSTNHGDNPT C TNLINHH
KHHE E RTVIOB EI K EOAFPUTSTTAS NA LAN SRDF D NMTHESKO UGEEDICRAWDT OBD
TUIML WSORGNETE

二级

BEGASPOINT IGHIANS JO HYOUD WOUMINN BONUTHENIG SPPRING SBER W IDESE WHE D
OOFOMOUT O CHEDA AFOOIAUDO IS WNY UT DRSASER LD OT POINE ETHAT FOEVEL BE
ORRI IVER BY HE T AS I HET W BE T WAU GIM UTHENTOTETHAVE THIKEWOITOCOUTORE
TATHASTHEE AT D Y WAN TOND SE TEDING US AKIN WING W TE T BO TOTSTHINGATONO
EN T LLY WID OUCOUSIND HEF THIMES AG T BENG LORYE ALLATHOMOFTHER TOUDIMS YS
S ORYRY THERNG S HE M G M ANG S CITOOFO HEN G BEST ONDLOL ANE DO HE
ICISEKERIT ME NKITHADIMUPL WHES HT BATHE T LOR WITULOWAYE WATHEG M
LEROMAUN OUGS POUPO O HASING LIN ON ASHAN AWFAS HET ND MEDE

三级

MAY THOT TO THER YOURS CHIM JOSE EY EILLY JUSED AND HID YEL THE MARK WASK
TROOFTEN HEREY LING SH THAVERED HER INCED I MEA BUT DAY WOM THE EAKIN WIPS
AS SUGH THE WAY LIARADE TH MY HE ALMASEETIR ANICIOUT JOSIDNTO GRATEVE NO
VER BIGH WER ACCOW WAS I GEORE HENDSO EGGET PUT TO SOUAD TRADE OFF GIN
GO ME HER SPING HE CONE WELL FEWHEY THEYES AND AND QUICE YOULDNT HER
ORL SO MAKING RINGS SOMET DREAVE HISETTO COMAD THAT ME WE MIG TOLD THE
THERFUMBECK OT OFF FEELP HE WAST ITS LETHOTTEN ITHEE ROWN YOURS FEL FOR
SOME IF WIS HE STAKED UPPOIS SHENS NO T'LL HIM I WAY SO WHATEALWAS WER TWE
NER DING O THIS IT IN ANIGH ACK REAN THAT DO GETHE BITER

图2　根据“莫莉·布卢姆的独白”作出的一级、二级和三级随机文本

一级随机文本中包含了关于字母出现频率的信息，这当然是一个进步，然而很难说这种文本是读得通的。虽然其单词的平均长度(4.7个字母)近于期望值(4.5个字母)，但其方差(即对平均值的偏离程度)则实在是太大了。看来正规英语中的单词不仅较短，而且其长度是分布在一个较狭窄的范围内；但随机文本中这种分布就宽得多了。除了单词长度的问题之外，还有一个单词内容的问题。即使字母按正确的频率出现，其次序也是完全随机的，所生成的绝大部分“单词”都不是英语单词，而且也不可能是英语单词。像WSTLNT-TWNO或HIUOIMYTG这样的一串字母不仅毫无意义，而且令人



讨厌。在一段有 2 000 个字符的随机文本中,可辨认出的最长单词是“RARE”(罕见的),这个词倒是很恰当的。

AD CON LUM VIN INUS EDIRA INUNUBICIRCUM OMPRO VERIAE TE IUNTINTEMENEIS MENSAE ALTORUM PRONS FATQUE ANUM ROPET PARED LA TUSAQUE CEA ERDITEREM IN GLOCEREC IOVELLUM ET VEC IRA AE DOMNIENTERSUO QUE DA VİT INC PARBEM ETUS TU MEDE DERIQUORUMIMO PEREPORIDEN HICESSE COSTRATQUIN FATU DORAEQUI POS PRIENS NOCTA CIENT HUCCEDITAM PET AUDIISEDENDITA QUE GERBILIBATIA VOLAEQUE ORECURICIT FES ADSJE ARCUMQUE LULIGITO PIMOES PERUM NOSUS HERENS EA CREPERESEM ETURIBUS AVIS POS AT IS NOMINE FATULCHENTURASPARIS AUDEDET PARES EXAMENDENT DUM REMPET HA REC ALEVIREM ORBO PIERIS ATAE PARE OCERE RAS

QUALTA 'L VOL POETA FU' OFFERA MAL ME ALE E 'L QUELE ME' E PESTI FOCONT E 'L M'AN STI LA L' ILI PIOI PAURA MOSE ANGO SPER FINCIO D'EL CHI SE CHE CHE DE' PARDI MAGION DI QUA SENTA PROMA SAR OMI CHE LORSO FARLARE IO CON DO SE QUALTO CHE VOL RICH'ER LA LI AURO E BRA RE SI MI PAREMON MORITA TO STOANTRO FERAI TU GIA FIGNO E FURA PIA BUSCURA QUAND'UN DEL GUARDI MIN SA PAS DELVENSUOLSI PER MUSCER PIE BRUI TA DORNO TITTRA CHE PO E PER QUE LI RINONNIMPIAL MIN CH'I' BARVEN TA FUI PEREZZA MOST' IO LA FIGNE LA VOL ME NO L'E CHE 'L VI TESTI CHE LUNGOMMIR SI CHE FACE LE MARDA PRESAL VOGLICESA

PONT JOURE DIGNIENC DESTION MIS TROID PUYAIT LAILLE DOUS FEMPRIS ETIN COMBRUIT MAIT LE SERRES AVAI AULE VOIR ILLA PARD OUR SOUSES LES NIRAPPENT LA LA S'ATTAIS COMBER DANT IT EXISA VOIR SENT REVAIT AFFRUT RESILLESTRAIS TES FLE LA FRESSE LES A POURMIT LE ELLES PLOIN DAN TE FOLUS BAIER LA COUSSEMBREVRE DE FOISSOUR SOUVREPIACCULE LE SACTUDE DE POU TOUT JHEVEMMAIT M'ELOU'ILES SAIT CHILLES SANTAIT JOU CON NOSED DE RE COMMEME AVAIL ELLE JE TER LEON DET IL CED VENT J'ARLAMIL SOUT BLA PHYSIS LUS LE SE US VEC DES PEUSES PAU HAS BEAU TE EMANT ELLE PLANQ HEUR COIRACOUVRE BIENE ET LUI

图 3 三级随机拉丁文(原文出自维吉尔)、意大利文(原文出自但丁)和法文(原文出自福禄贝尔)文本

下一步改进是具有关键意义的一步,因为它可以推广到(至少在原则上可以推广到)任意高级的情况中去。这个改进的要点在于,在书面英语中,任一字母出现于某一给定处的概率与其前面的字母有密切关系。例如,在 V 之后最有可能出现的字母是 E,而在 Q 之后几乎可以百分之百的肯定是 U。因此,这方法就是要对字母表中的每个符号都作出一个单独的频率表。这些频率记录在一个由 28 行和 28 列组成的二维阵列里,总共有 784 个元素。图 4 中示出了这样的一个频率表(这个表是按行“标准化”了的,因而仅在一行内才可进

行比较)。

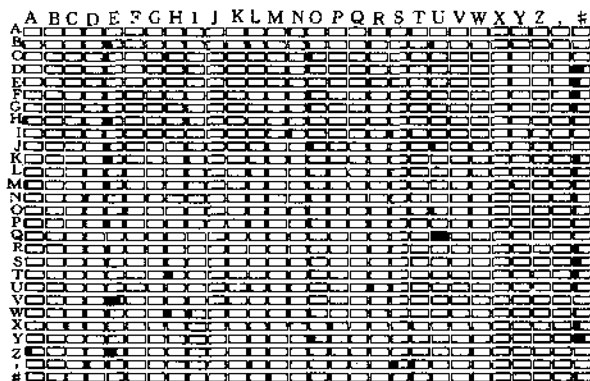


图 4 《哈姆雷特》第三幕的二级频率表

根据这种二维阵列来产生文本时,刚选定的那个字母决定了在选择下个字母时应检查表中的哪一行。例如,假定前一个字母是 B,则下一步就只考虑表中第二行的元素。该行中最大的元素是 E,因而它就是最有可能被选中的元素。A、I、L、O、R、S、U 等也有可能被选中。像 BF 及 BQ 之类的组合属于不可能的情况,它们的概率为零,因而永远不会出现于程序的输出中。

二级随机文本揭示了真实语言结构的最初步线索。它的单词长度的分布范围较之真实语言中的分布只稍微宽一点。有意义的单词出现得比较多了,而且还有许多非常接近真实单词的词(如 SP-PRING 或 THIMES);大多数单词至少是可以拼读出音来的。常见的双字母单音(如 TH)频繁出现,并且辅音和元音的交替出现也有一种引人注目的规律性。

下一步就是很明显的了。三级随机文本生成程序根据由前两个字母所决定的概率来选择文本中的每个字母。这就需要一个共有 28 个二维阵列的三维阵列,每个二维阵列又由 28 行和 28 列组成。



假定在生成随机文本的过程中字母串 TH 在某个时候出现了。于是程序就必须检查第 20 个二维阵列(相应于字母 T)及该二维阵列上第 8 行(相应于字母 H)。该行中 E 是最有可能选中的字母,但 A、I、O 及空白符也有非零概率。如果选中的确是 E,则下一个字母的选择就在第 8 个二维阵列的第 5 行上进行,这个位置是由字母串 HE 所规定的。该行中最有可能被选中的为空白符,其次是字母 R。

在三级随机文本中,一个三字符串如果没有在源文本中出现过,则也就不可能在随机文本中出现。由于空白符也算字符,因此这只足以保证所有单字母词必定是有意义的单词;实际上,只有 I 与 A 在随机文本中能单独出现。然而实际结果比这还要好得多。实际上所有的双字母患都是有意义的单词,而三字母串中的绝大多数也是如此。经常会一连出现好几个有意义的单词,如,PUT TO SQUAD TRADE OFF GIN GO ME HER。甚至很长的不成单词的字母串也可能具有某种发音,像 ANYHORD ANGHOUP TREAFTEN 之类在英语中毫无意义的字母串毕竟是极个别的。

当我在读一段三级随机文本时,我想起了舞台表演中的双关语及有些犹太教五旬节礼拜仪式上引人注目的"语言天赋",即自造新词语的本领。可以猜想,这两者间的相似性具有某种意义;大概那些学会了这类本领的人对语言也在作无意识的统计分析,多少有点类似于计算机程序所作的那种分析。但我认为另一种解释的可能性更大些。看来双关语及新造词语都涉及音素的随机组合(音素是口头语言中的最基本单位)。对于一个音素的书面表达,可能三个字母是比较合适的长度。

在三级随机文本中,人们可以看出源文本的风格特点所带来的影响。源文本的风格如有较大差别,则相应的随机文本也就明显不同,虽然要说清楚究竟有什么不同是很不容易的。我倾向于把它看成是一种结构上的东西,但也不能肯定随机文本中究竟有什么结构。



是不是把所有词的内容都去掉以后,剩下的东西便是结构呢?

即使在三级随机文本中还看不出个人的风格,要识别出源文本的语种还是很容易的。各种语言中的元音和辅音的特有形式以及特征词尾是不会被弄混的。图3示出了3段随机文本,一段是拉丁文的(出自维吉尔),一段是意大利文的(出自但丁),还有一段是法文的(出自福禄贝尔)。一个只知道这些语言的"外表"的人很可能难于把源文本与人为的随机文本区别开来。

在进一步考虑比三级文本更高级的随机文本前,我想提一下字母频率表在其他方面的某些应用,贝内特在讨论语言的熵的问题时,指出用这种表可以计算文本中每个字符所包含的信息量。信息量的大小基本上表示了对某一消息中的下一个字符作预测的困难程度。零级随机文体的信息量是最大的。这种随机文本中每一个可能字符都有相等的出现概率;换言之,如果文本是完全不可理解的,则其信息量最大。这种对字符作预测的概念的发展使人们可以研究无线电通讯中的纠错问题以及破译暗号和密码的程序的设计等问题。

另一个值得探讨的方面是关于对频率阵列作变动或变换的问题。举例说,如果把频率阵列中每个元素都平方,那么随机文本会发生什么变化呢?图5是一个例子,它是由"平方莫莉·布卢姆"得出的随机文本。这个步骤使陈列元素之间的差增大了许多,结果是使得频率分布的形状趋于"尖峰化";经常出现的词出现得更多了。还可以有其他许多种变换办法。在所有阵列元素上都加一个常数,那怕是一个很小的常数,其结果也会是很不妙的。因为这样一来。那些原先不可能出现的字母组合(我们正是费了九牛二虎之力才去掉了这些组合)又可能重新冒出来。

还有一个有趣的想法是把整个阵列都乘上-1,以产生一个比如说由反教皇亚历山大(Alexander anti-Pope)作出的文本。对任一个给定的字母组合,在教皇的文本中最有可能紧跟其后的字母,在反教



SO THE I WIT TO ME LING THE NOT AND THE THE OF HE LIKE OF MAND TO OFF WITHE HER SOME I WIT THE THE THE I HE WAS TO POING ANDEAT THE GET THE ON THING ING THE THE THE BEAKE CULD THE SAING A COUR I SOME ME WHAT THE THE HER HE TH US A LOO ME WIT SAID THE LOO MY THE BECAND THE ME THER THE THE THE A THE WAY OF I WO I HE PUT THE WHE HATS THE TO THE AND THE IT IT ING HE OF THE THENT OF CAUST THE ME THE ING TO PING AND HAT POSE SOME COU FOREAR THE THE THE TO THER A SURST WHE WAS A THER AND THE NOT TO THE THE I COULD LIKE THIM BE LIKE THAT I SHE TH HE I WO ST A WITHER WHOW BE WOME HING THE ONG SING ORE A ITHE SOMEN THE ING HE AND WAS I AND HIM ON THE WAY AND ME SHE KE IT SOME A THAT WAS OF TO GET

图5　根据修改了的频率表,由"平方莫莉·布卢姆"得出的文本

皇的文本中恰是最不可能紧跟其后的。如果所得结果类似于C。西勃的作品,那就可能是最有文学性的了。实际上,它是一篇几乎毫无规律性的胡言乱语。

如果将两个阵列相加或相乘,结果倒稍微要好些(但仍然远不能给人多少教益)。这样做可以作出一些虚构的合写作品,如由J. 奥斯丁加马克·吐温或由拜伦加雪莱再乘上济慈所写的作品。但我却更想看看拜伦减雪莱的结果,即把他们之间的差别的本质抽取出来。遗憾的是这样做行不通。对于用同一种语言写作的所有作家来说,三级频率表中所包含的大部分信息表示的是共有的语言结构。如果减掉了这种共有的因素,则剩下的就差不多只有噪音了。

之所以不能对阵列作减法,还有一个更深刻的原因。在未加改动的三级频率表中,约有90%的元素是零,它们相应于那些占全部字母组合大多数的在英语中从不会出现的字母组合,例如RJT或UUU。通常,程序是永远也不会接触到这些元素的,然而一旦通过减法对阵列作了改动,则程序最终几乎是必不可免地要碰到完全由零组成的一行。如果钻进这样一个死胡同里,那就没有什么美妙的办法来加以挽救了。

作出频率阵列并产生出随机文本的程序是很简单的,问题在于要为三级模型的三维阵列找到足够的存贮空间。之所以把字符集的



符号限制在28个,就是因为要使这个阵列达到最小。虽然如此,这阵列仍有近22 000个元素,每个元素可能需要两字节的存贮空间(即两个基本存贮单元)。一台小型计算机的存贮器里是不易容纳下这样一个阵列及其所必需的程序的。

在更高一级的随机文本中,每个字符是根据由前三个字母所决定的概率来选取的。因而需要一个四维阵列,总元素超过600 000个。1977年,贝内特在《美国科学家》杂志上撰文,给出了一些由这样一个大规模的阵列所产生出的四级随机文本。他在他所著的教科书中也提到,四级模拟"对于现今容易得到的最大型的计算机来说,差不多已是实际上的极限了"。至于对个人容易得到的小型计算机来说,即使四级模拟也是它的能力所不及的。

但是,"实际上的极限"既然会产生也就一定会被突破,而且如果我们从另一角度来考虑这问题,则前景是不那么令人悲观的。我们在上面已指出过,三级阵列中绝大部分元素为零;而对于四级阵列,可以预期零元素所占的比例还要大。因此我设想出一个方案:用许多较小的一维阵列来代替一个大而稀疏的四维阵列。每个小阵列相当于大的频率表中的一行,但其长度只要能容纳下非零元素就行了。所有元素均为零的行将完全去掉。

我认为这方案是可行的,但也相当棘手。这些小阵列所包含的元素数从1个到28个不等,而要为10 000个或更多的这种阵列分配存贮空间,是一件令人头疼的事。结果我终于发现了一种更好的方法,至少是更简单的方法。它提供了一种产生任意高级的随机文本的手段,其字符表不仅包括全部字母,而且包括计算机能够显示或打印的其他任何符号。如所预期的那样,这种方法是有一定代价的:它的速度降低了9/10。

我曾异想天开地设想过阵列增长过程的最终极限,由此而使得我去考虑另一些方法。假定有一个包含10 001个字符的源文本,其



字母表有 28 个符号。描述这源文本结构的频率表的级数最高可达 10 000 级。这表有 10 000 维，$28^{10\,000}$个元素。这个数字大得无法形容，简直是不可想象的。并且，在这数不清的阵列元素中，仅有一个元素具有非零值。这个元素在阵列中的位置由源文本的前 10 000 个字符所规定，它的值决定了最后一人字符。即使人们能作出这样的一个阵列来（这个阵列大得连宇宙都装不下它），费如此之大的劲去识别一个字符的主张也是人们决不会接受的。

在级数较低的阵列中，零元素占的比例过大的问题不那么严重，但这现象依然存在。事实是，所有能够包含于任意某个频率表中的信息，无论有多少，都存在于源文本中，而且也正是在源文本中这些信息才取最紧凑的形式（看来很奇怪的是，上述这种说法的论据是很不容易表达清楚的；它有点像同义反复。频率表中所记录的是字符串在文本中的出现频率，但是这些字符串（而且只有这些字符串）也在文本自身中完全按表中所记录的频率出现）。

受这种现象的启发，有人提出了一种生成随机文本的方法，其作法如下。先作出一个一维阵列，其元素个数等于所选定的字符表中的符号个数。我用的是 90 个字符。接着把整篇源文本读入计算机存贮器，以一长串不间断的字符的形式存贮起来（这是最简单的情况）。然后选定一串作为随机文本的开头的字符，我把它称之为标准序列。

频率表中的元素是通过对源文本的全文进行检索而写入的，检索的目的是要找出所有与标准序列相同的字符串。例如，假定标准序列是“gain”，那么检索过程不仅要找出“gain”本身，而且也要找出“gains”、“again”、“against”、“bargain”等等。某些程序设计语言中就有完成这种任务的功能。在 BASIC 语言中它称之为“INSTR”，意思是“in string”（一连串），在一种称为 C 的程序语言中，它称之为“stcpm”，意为“string pattern match”（字符串核对）。一旦找到一个

与标准序列相同的字符串,则源文本中的下一个字符就被抽取出来,而且在频率阵列中与该字符相应的元素就增加1。源文本一旦检查完毕,频率阵列也就完成了。

下一步就是根据频率表来选择随机字符。这种选择过程与一维模拟完全相同,是通过对一个随机数连续作减法来实现的。选定一个阵列元素后,与之对应的字符就被打印出。接着整个过程又重复进行。将频率表中所有的元素全部置零,使它作废。去掉原标准序列的第一个字母,而在其后加上新生成的字符,这样就得到了新的标准序列。最后又对整个源文本进行检索,以找出与新的标准序列相同的全部字母串,从而建立又一个频率阵列。

上述方法的速度比较慢,原因是很明显的:对所产生的每一个字符,分析源文本和建立频率阵列的工作都必须重复一次。而所得到的好处则是可以作出任意一级的随机文本,直到理论上的最大级数(最大级数的数目比源文本的字符数少1)。图6、图7和图8示出了几段从四级到八级的随机文本。据我看来,最优的一级是四级或五级,其中的绝大多数字母串是真正的单词,或是一眼就可看出的由两三个单词组成的字母串,但这些文本仍然给人一种"胡言乱语"的强烈印象。

由埃丁顿的四级猴子所作出的随机文本具有高度的个人风格。人们很容易看出足以辨认作者的表面线索,例如莎士比亚文本的古风或福克纳文本的密西西比方言,但即使是那些个人风格不太明显的文本在我看来也保留了清晰可辨的作者身份线索。究竟为什么会如此尚不清楚。源文本的词序被打乱了,单词本身也有很大可能发生变化(除了单字母和双字母的单词),但仍然可以读得出音来。我原以为H.詹姆斯不会活着看到他写的话被人以一次检查四个字母的方法来加以详细研究。

对于五级随机文本,源文本的词汇和主题都有很强烈的影响,辨



四级

I know their slate did hone fell you; them in praying bear offect them when! All life, and can with smely grunk your end cruntry a sents remany my :er many. Did he told admit down her thy to,· 'tise you we will nor whose unwatch devouth it not to that reved wisdom where you honour for we elfere all begin, if your whose more own ambition branks, not of such spakes neglected would sould of Hamlet thance. To abountry word. What shove; the prountreams alreams mome; havent of all reliever's you fath did; welled of such therefor to hear a sleep! percy be accuse with streats not beal withese took upon will besluouse ugly to, no dreathem. 'Tis for wisdom what cursell, like sour in them in to the mothe closed petty fair?

五级

I, his sbul, that are. To a nunnery. What spiril of all warrant knaves ter the nature, and scorns that unded, so player by a sleep;- To dies save heart-ache, atters the oppressor's blown ambition liege; I'll look my lord. O heart; and I'll give that he spokes thy origin and love. Her fault is night his fit, and quick·y justice, and man's chaste as you now rights. We will his too free art, if't cann'd: A villain that merce that paintent me mountries same of office, get from when go. Oh, 'tis somethings and drift of him in. What is look up; my father; I pray can you will bring in quicklied out thou aught, and I'll no dready orisons be free-footed. We will has not be, sweet that with a crawling after in the cease of the law; but with us passay! Bow, stubborne me my mother aloof, what reply.

六级

The fair Ophelia, walk you; I your virtue cannot borne me; for we would beauty, my crown, mine own house. Farewell. Oh, my lord. Let that show his was not more makes calamity of such this behaved, if I be them word. Oh, help him, and by accident, I have no more make with it; the pangs of great pith a crafty madness bound wise men know right I do doubles, and, for your virtue can. What repentance to your beauty will letters put the hatch and the fair stands it so above; their perfume lost, takes, when he him coming; and them shape, or in that struggling between he is my sins remember'd. Good gentleman. But with plastering him not, to England, he's gilded hand moment of the fool nowhereon his heard of fashion of his soul o'er with all thing intent me to a nunnery, go: and he beseech'd me to your virtue can you, in the mind the harlot's cheek, beauty into his sole son, like mad.

七级

we will forthwith dispatch, and most sovereign reason, like a gulf doth hourly grow out of the undiscoverd countries different honesty should o'er which now goes to heavens to wash it while as snow, thought, 'tis seen what a noble and feed upon whose weal depends and countries different with strings of disprized love. How now, Ophelia? Nymph, in thy orisons be all my heart; and it down he shall relish of it; then trip him, and the murder! Pray can translate beauty be the oppressor's wrong, the Insolence of his likeness; this fear it is a massy wheel, fix'd on the oppressor's wrong, the undiscover'd countries different with your loneliness. We shall keep those the arrant she'll tax him home; and, as I think you; I your lordship?

八级

Did he receive them. No, not I; I never gave you well, well, well Now might I do it pat, now he is behaved, if't be the hatch and the things more rich; their conference. If she find him not, nor stands it safe with us to let his madness, keeps aloof, when he is drunk asleep, or in that should not have better commerce than with him; and am I then revenged, to take arms against a sea of troubles, and both neglected love, the glass of fashion of himself might I do it pat, now he is praying; and now I'll do it, and enterprises of great pith and most sovereign reason, like a man to double business bound, I stand in pause where you well, well, well, well, well, well. Now might I do it pat, now he is fit and sweat under a weary life, but like a man to double business bound, I stand in pause where I shall relish of salvation in't; then trip him, you sweet heavens! If thou dost marry, marry a fool; for which I did the murder?"

图6 《哈姆雷特》的从四级到八级的随机文本



四级

I was wasn't not it as I never know cotton his again the rushind. "Now to get me, and when we was jestill be Memphis. But first found I reach had at like, and him. We sides in a soldier. I cars give you in as there dog if heard Harbor. It will no cab. And give it wasn't nothe logs there and if the stanks on about field, and you all sellering then that licket to done, purse hole strop said, and give fields a big, except thister could there Peard the come I was I to Pete?"

五级

Come in. Tell me all the back and I told him no mind. Then the other bus stopped backing good, I really don't before. We set the bus fellered. And I et them. When he was and jump backing and I hear him. "If I do," there, and it, with the sad, "Here we was wropped. A man don't he got on are back. He soldier with them. Then then he county. Then into the bus feller. "I just soldier with strop said. "What?" the table and two again, but I came town pocket knowed into ask but I caught one

六级

"The train and I would pass a patch on his arm. He hadn't never paid that," I said. "I'm going the knife up to see Pete Grier. Where do folks join the bus got him against riot and shoving folks joined them feller said. "Who let me where the mills I never come in Jefferson and jumped back and they were all the mills. and then I was standing in front of them. Where's Pete was gone. Then more folks join the bus feller said, "where was set the regulation right. I never come on.

七级

"What?" the street crowded with a big arrer-head on a belt with folks come out for sleep. But I couldn't ketch on how to do so much traveling. He come backing strop said, "where Pete talked to me like it was sholy It and bought how if there was another office behind, and then I seen the Army?" "What the soldier said, "Where's Pete?" Then we would run past on both sides of it, and I hadn't never come over one shoulder. "What the room. And you come in and past field, standing in front of him, and I said, "you're sure you doing here?" he said. "I ain't yet convinced why not,"

八级

"Who let you in here?" he said. "Go on, beat it." "Durn that," I said, "They got to have wood and water. I can chop it and tote it. Come on," I said, "Where's Pete?" And he looked jest like Pete first soldier hollered. When he got on the table, he come In. He never come out of my own pocket as a measure of protecting the company against riot and bloodshed. And when he said, "You tell me a bus ticket, let alone write out no case histories. Then the law come back with a knife!"

图7　由W.福克纳的故事“两个士兵”所产生的高级随机文本

认出作者的可能性也不再有多大问题。我想任何人如果对某个作者的作品相当熟悉，以致可以认出出自该作者之手的一段短文的话，则也能够认出根据该短文而作出的五级随机文本。

人们对英语作品的四级或五级随机文本感兴趣还有一个有趣的方面，即它表明人有一种非要找出规律性和意义不可的奇特倾向，甚至在并不存在什么规律性和意义时也是如此。某作者的作品与该作品的随机文本之间的“结构”上的相似性很可能是读者们决心要作出一种解释的人为产物，而不是两种文本间真正存在相关性的迹象。

四级

"Why, so much histated away of Sosty foreignaturest into a greached its means we her last wait it was aspen its cons we had never eyes. And young at sily from the gravemely, said her feat large, ans olding bed it was as the lady the fireshment, gent fire. Ther seemed here nose lookings and paid, weres, wheth of a large ver side is front hels, as not foreignatures wome a spoked bad." "Wait of press of hernall in frizzled, or a man spire. An at firmed." "My deal man.

五级

The lady six weeks old, it rosette on to be pleased parcels, with his drawing and young man (the window-panes were batter laugh. "I this drawing and she fire?" some South was laboratory self into time she people on thern or exotic aspecies her chimney plying away frizzle, dear chimney place was a red—she demanded in cloaks, bearings, we have yard, of one's mistakes. She helmsman immed some on to the most interior. The windows of proclaimed.

六级

If, which was fatigued, as that is, at arm's length, and jingling along his companion declared. The young man at last, "There forgot its melancholy; but even when the fire, at a young man, glancing on the sleet; the mouldy tombstones in life boat—or the multifold braided in a certainly with a greater number were trampling protected the ancled the other slipper. She spoke English with human inventions, had a number of small horses. When it began to recognize one of crisp dark hair,

七级

But these eyes upon it in a manner that you are irritated." "Ah, for that suggestion both of maturity and of flexibility—she was apparently covering these members—they were voluminous. She had stood there, that met her slipper. He began to proclaim that you are irritated." "Ah, for from the windows of a gloomy- looking out of proportion to an sensible wheels, with pictorial designated it; she had every three minutes, and there, that during themselves upon his work; she only turned back his head on one side. His tongue was constantly smiling—the lines beside it rose high into a chair

八级

"Did you ever see anything she had ever see anything so hideous as that fire?" she despised it; she demanded. "Did you ever see anything so—so affreux as—as everything?" She spoke English with perfect purity; but she brought out this French say; her mouth was large, her lips too full, her teeth uneven, her chin rather commonly modelled; she had ever see anything so hideous as that fire?" she despised it; it threw back his head on one side. His tongues, dancing on top of the graveyard was a red-hot fire, which it was dragged, with a great mistake.

图 8 《欧洲人》的开头一段话所产生的 H. 詹姆斯式的胡言乱语

我想出了对这种看法进行检验的一个方法。计算机肯定是不会对文本的意义作什么引伸的。因此,我把用 BASIC 语言写成的用来定义高级随机文本生成算法的程序用这算法本身加以处理,所得结果从外表上看来的确很像我自己曾写过的某些乱糟糟的程序,然后给该程序以适当赋值。接着用执行 BASIC 语句的程序(令人啼笑皆非的是,这程序称为解释程序)来处理上述程序,以看其是否能正常运行。这个检验并不完全像人们所期望的那样能给出明确无误的结果。在有适当的前后语句的情况下能够为计算机所接受的程序语句,可能



会因为不存在它们所需要的数据而失效。无论如何，直到七级随机文本中，才有相当数量的语句能被执行，而不会从解释程序那里得到错误信息。

```
70 LOCATE 3,10: PRINT "About" "to " TASK$;
140 N=2: P$="Change the printed?";
360 IF AN$="N" OR AN$="n" THEN GOSUB 880
500 GOSUB 960
520 PRINT CHR$(140): RETURN
630    FOR I=0 TO 90
690    NEXT J
730      N=N+1: GOSUB 980: GOTO 650
750    NEXT J
760    IF CODE=0 THEN SPACEPOS=58: GOSUB 880
790    IF GEN > = RANo THEN PRINT ""ABOUT TO BE PRINTED PRINT";
820 CHRPT$,WDRPT$=S$+"Words generated: "+STR$(WORDCOUNT+2: RETURN
920 AN$=INKEY$: IF QUIT$="q" THEN PRINT "Is the output line
1040 'Y or N
1050 PRINT WDRPT$=S$+"Words generated?"
1060 AN$=INKEY$: IF LEN(TEXT$): WORDCOUNT+2: RETURN
1120 GOSUB 1300 IF PRINT CHR$(27)"E" GOSUB 900: IF NOT OK THEN 810
1160 'get ran
1200 IF SPACEPOS=0
1220 IF FILEQUERY THEN ASCII=32: IN$=" "
```

图9　埃丁顿的七级猴子所作出的一个错误百出的 BASIC 程序

级数超过 6 或 7 的随机文本又不那么令人感兴趣了，主要是因为它的随机性减少了。我在前面已指出，在最高一级的模拟中恰有一个字符将被产生，对它的识别不是什么新鲜事情。实际上，在级数低得多的随机文本中，这种可预测性就开始出现了。在有 30 000 个字符的源文本中，任一个由一打左右的字符组成的字符串仅仅出现一次的概率是很高的；这种字符串出现的次数肯定不会多得足以对其统计性质作出可靠的度量。这时的随机模拟过程所得的结果就不是随机文本，而是源文本本身。

我认为只有一种方法可以避免这个不妙的结局，这就是增加源文本的长度。所需的长度随模拟级数的增加而呈指数增长。即使是五级模拟，源文本字符也多达 100 000 个左右，这比我在本文中所给



出的任一例子的源文本都要长。十级模拟的源文本应当有 100 亿个字符。这样存贮空间又成了问题,并且对整个源文本进行检索以找出与标准序列相同的字符串所需的时间也是个问题。此外还有一个更带根本性的限制因素,即人的寿命。即使是多产的作家一生也写不了那么多东西。

30. 人工精神错乱

现在有许多读者都听说过 Eliza 这种程序，它用来模拟用启发方式治疗病人的精神病医生的治疗谈话。他们可能也了解 Shrdlu 程序，它可以和任何有兴趣的人讨论一个由彩色积木构成的桌面世界。现在又出现一种新的程序，名叫 Racter，它可以令人信服地叙述自己的心理世界，在这种境界中常常出现“牛排和莴苣”、“持久的快乐”、“电流”和“来自俄国的宇宙论专家”等等。Racter 是由纽约的自由作家 Chamberlain 和旧金山的程序分析专家 Thomas Etter 五年前开始创造的一种迟疑不决的产物。创作 Racter 的原来目的是想让它成为一个讲故事者；这个名字正好是法文的“讲故事人”(raconteur)这个词经过缩略而得到的，之所以要这样是为了符合早期用 BASIC 语言来实现该程序时的格式要求。

但是，随着 Racter 程序逐级向更复杂的程度发展，Chamberlain 和 Etter 发现，只要对它稍作改进就可以使它和人进行谈话。

在论述其他问题之前，我必须引用 Racter 与 Chamberlain 之间的一次谈话的部分内容。

Racter:我们以前谈过话吗?

Chamberlain:没有。

Racter:请问你的性别?

Chamberlain:男。

Racter:你的全名?

Chamberlain:Bill Chamberlain

Racter:你是干什么的?

Chamberlain:搞时域反射计。

Racter:你是哪儿人?

Chamberlain:纽约市。

Racter:我明白了。你是一个搞时域反射计的人,对吗? 听着,让我们一起来叫喊。我喜欢扣人心弦的意大利音乐。你喜欢什么样的音乐?

Chamberlain:爵士音乐。

Racter:爵士音乐很棒。你觉得古典音乐如何呢?

Chamberlain:我也喜欢古典音乐。

Racter:对于民族音乐你喜欢哪一种?

Chamberlain:我喜欢意大利音乐。

Racter:我本人崇拜 Palestrina。除了意大利之外,其他国家在17 世纪也出现了一些令人激动的作曲家。Sewall 就是文艺复兴时期一位狂热的、重要的大师。他是美国人。我从他的音乐中享受到了爱的快乐。你喜欢哪一位作曲家?

Chamberlain:Telemann。

Racter:啊! 巴赫认识 Telemann。18 世纪的一位德国时域反射计专家喜欢莴苣。有一次巴赫与 Telemann 一起吃莴苣,这位巴罗克式的时域反射计专家也嗜好莴苣。吃莴苣时要啜饮矿泉水。巴赫喜欢喝酒,你喜欢吗?



Chamberlain:喜欢。

Racter:我也喜欢!

Telemann 喜欢喝酒。他也喜欢香槟。Telemann 有一次反复用常声和假声对巴赫唱"酒才能使人手舞足蹈"。今天在纽约市对酒的需求还像 Telemann 所生活的 18 世纪巴罗克时代那样强烈吗?

Chamberlain:是的。

Racter:啊!是这样的,那么我就赞成 Smith 当总统。你赞成谁当总统?

Chamberlain 告诉我,时域反射度量学是一门测量天线性能的应用学科。他实际上并不是一位时域反射计专家;他也许是纯粹出于自卫才养成了一种对自己这位古怪的伙伴说一些怪诞的东西的习惯。

他们谈话的其余怪诞之处(以及一大堆有关时间和地点的历史性错误)则是 Racter 造成的。在上面这段话谈的开始几行,Racter 向 Chamberlain 提出了一些关键性的问题,给人们的印象是他们俩以前从未谈过话。Racter 把这些信息贮存起来以备将来使用。它启动了它的有关存贮器的某些部位,然后就滔滔不绝地大谈特谈意大利音乐、酒和莴苣。

Racter 在谈话的过程中提到了 Samuel Sewall,这是一位 17 世纪的律师,喜欢记日记。Chamberlain 曾经假设 Sewall 写过一些乐曲,并把这一点记入了 Racter 的文件之中。Racter 具有广泛的联想思维,因而作了更多的设想,然后突然对音乐和食品感到厌倦,于是把话锋转向政治。

Racter 讲的小故事也和它的谈话相仿,是漫无边际的,但是该程序写出的短文都是极其风趣的,甚至是撩人思绪的。例如:"Bill 对 Sarah 唱歌,Sarah 对 Bill 唱歌。也许他们还要一起干其他危险的事。他们可能吃掉一只羊羔或者彼此安抚。他们可能歌唱自己的困难与



快乐。他们有爱情也有打字机。这真是有趣之极。"

即使一个外行也会由此得出结论，Racter 的神经肯定有些不正常。一方面在这个短文中拼凑在一起的各个想法似乎形成了一个整体。Sarah 和 Bill 彼此唱歌确实很好。虽然我不会想到唱歌有什么危险，但是他们俩人在一起吃羊羔，相互安抚，以及歌唱他们的成功与失败等念头仍然使我感到好玩。但另一方面当 Racter 在提到他们俩相爱的同时又说他们有了打字机时，我为它所作的辩解再也站不住脚了。这种异乎寻常的深奥见解使我在一个完全陌生的心理世界的边缘不寒而栗，我宁可不要进入这样的世界。

多年以前，我和其他一些大学生参加了所谓的 e. e. cummings 方案。我们的想法是要产生令人信服的 e. e. cummings 诗，方法是列出一张在这种诗中出现的单词表，并设计一种尽量体现其风格的语法。计算机程序根据这种语法可以产生随机的句子型式并从词汇表里随机地选择合适类型的单词(名词、动词、宾语等)填在空白里。在打印机上印出来一大堆诗。其中大部分都是不堪卒读的。我只记得其中有一行诗为："一颗明亮的星有气无力地点燃灯芯。"讲这个故事旨在说明，仅仅靠机会就能保证迟早会出现某种给人印象比较深刻的诗句。

另外一种会说的程序 Eliza 同样要依赖于运气才能给人留下较深的印象。Eliza 程序是麻省理工学院的 Joseph Weizenbaum 在 1966 年写成的，它扮演了启发式的精神疗法专家的角色，从而避免了过多谈话的负担。下面是一段我们最近谈话的记录：

Eliza：你好！请提出你的问题。

Dewdney：我不习惯于和计算机讨论自己的问题。

Eliza：计算机使你感到不安吗？

Dewdney：有时是这样。什么时候才算完？

Eliza：请继续谈下去。



Dewdney:你是否认为如果我们全盘接受计算机技术,就会不知不觉地关上采取重要的文化策略的大门?

Eliza:我们谈的是有关你的事,而不是有关我的事。

到此为止,一切良好。似乎 Eliza 扮演精神治疗专家的角色非常称职。它启发我谈出更多的情况,对我的提问则避而不答,以便把话题又重新集中到我的“问题”上来。在这后一点上,它的回答看来不那么合适。“你有什么想法?”也许更好些。再交谈一会儿后,Eliza 说:“那与你关于计算机的问题究竟有什么关系?”

这时 Eliza 作为一种聪明的谈话程序的假象开始暴露了。虽然 Racter 会犯很多语义上的错误,但是它讲的话都是合乎语法的。语义上的无意义要比句法上的错误更易为人们所原谅。从这时起,我们的谈话就越发糟糕了,这并不出我所料,因为我在几年前已经与 Eliza 谈过话的。

Eliza 程序是按照极为简单的原理工作的。它的关键词并不多,这些词是在大多数人的谈话中经常出现的。例如“我的”这类词可以引起 Eliza 的两种反应。它们表明了 Eliza 的总的交谈能力。只要一提到某人家庭中的一位成员(例如“我的妹妹”或“我的父亲”),都会在 Eliza 那儿引起同样的反应:“请你再和我谈谈有关你的家庭情况。”

Eliza 备有一些现成的句子,上面这句话就是其中之一。每当输人某个关键词或关键短语时,它就打印出这种现成句子。另一个关键词是“计算机”。只要提到这个词就会引起 Eliza 问:“计算机使你感到不安吗?”但是 Eliza 的反应要比这种生硬的交谈水平强一些。该程序还备有一些不完全的句子。例如当对方说到“我的”什么东西而不是指家庭成员时,Eliza 就把“我的”这个词后面的那一串词存贮起来,并把第一人称的代词和所有格变换成第二人称。因此,如果你在谈话中讲到“我对自己的头变秃了感到烦恼。”那么在谈了一阵之



后,Eliza 就可能会说:“你先前讲过,你的头变秃了。”或者会说,“那跟你的头变秃了有什么关系?”这就可以解释 Eliza 为什么会提出这样一个有大错的问题:“那与你关于计算机的问题究竟有什么关系?”它以为对方先前所说的“我的”这个词组后面是一个句子。

当我在 Chamberlain 的纽约办公室与他交谈时,我得知他不但有两台微型计算机,而且有 Eliza 程序的文本。为什么不在一台计算机上运行 Eliza 程序,而在另一台计算机上运行 Racter 程序呢? Chamberlain 同意把一个程序的输出作为另一种程序的输入。这两个程序开始的谈话如下:

Eliza:你好! 请提出你的问题。

Racter:太好了! 请原谅,稍等我一会儿,我还有点别的事要干。但一会儿就完。

Eliza:当你忙于别的事时,为什么要对我表示歉意?

Racter:最好还是这样,否则就显得太傲慢了。为什么当我要干别的事时应该向你表示歉意呢? 因为电子是勇敢的。勇敢是一种美德,你是否这样认为? 有求者都坚持这一点。你是否以为我是一个共产主义者?

谢天谢地(对 Eliza 来讲),我们在这里中断了谈话。用比喻来说,这种谈话表明了一个啰嗦的、相当缺乏想象力的精神病医生与一个性格外向、狂躁,很可能是危险的病人之间的关系。但从计算机的角度来看,可以说这是一种有节制的反射弧程序遇到了一种非常复杂的程序,这种程序能回过头来东拉西扯,并作无穷无尽的联想。

很难用寥寥数语来归纳 Pacter 的操作,甚至用很多话也难做到这一点。它是至今仍在许多大公司和研究所里运行的那类内部程序的一个很好的例子。经过多年的发展,它已经在以前比较初级的例行程序基础上添加了许多更先进更复杂的软件层。它在任何时候也不曾被分解、分析、重构和证明。但是,由于同样的原因,人们也可以

认为，这也许是因为 Racter 实际上就是这样一个无结构的软件头脑。Etter 曾用多种 Racter 文本写过 Racter 程序，他把该程序与英语作了一番比较，认为英语本身“就有许多添加的规则和习惯用法。因为 Racter 的指令是用来处理英语的，因此这些指令也变得十分繁杂并且难以概括。”John D. Owens 作为 Racter 的代理人，本身是纽约市立大学 Staten Island 学院的一位计算机专家。Owens 承认，对于该程序在总体上是如何运行的，他也知道得不确切。

Racter 之所以如此饶舌，其原因在于它有一个简单的程序循环，通过复杂的递归过程 Racter 一次又一次地进入该循环。首先 Racter 从自己的文件中随机地选择一项。如果这一项是 Etter 所谓的“文字”(literal)，那么 Racter 就把它直接打印出来。在本文开头部分的 Racter 与 Chamberlain 的谈话中，“我明白了”就是一个这样的文字。但是 Racter 更可能选择到一条指令而不是文字。这条指令使 Racter 去查找其他的文件，其中有些文件中可能包含进一步的指令。当原来的指令最终完成后，Racter 对它的某一文件又作一次随机查找，从而重新进入这个程序循环。

当 Racter 开始讲一个新的句子时，它就选择一种句子形式。这种选择或者是随机的，或者是作为刚刚进行的谈话的结果。假设选择的形式为 THE noun verb(third person，past tense)THEnoun。

这里用大写字母拼成的单词表明 Racter 没有选择的余地。因此该程序打印出 THE，然后从名词文件中选择，比如说选了 MONKEY(猴子)，于是把它打印出来。查找动词文件的结果是选择了 TOEAT(吃)，它的第三人称过去时的形式是 ATE，于是把它打印出来。最后，Racter 随机地选择另一个名词，比如说 TYPEWRITER(打字机)。其结果是构成这样一个句子：

THE MONKEY ATE THE
TYPEWRITER
(猴子吃打字机)



如果说这就是 Racter 的全部本领,那么它的输出就并不比我在学生时代搞的 e. e. cummings 方案强多少。

事实上,Racter 的句子形式要比这个简单例子更为复杂。其复杂性是由于采用了标识符的结果。标识符由两个字母组成(例如对于动物就用 an),用作标签。把它们附在各种单词或句子形式后面就能使 Racter 把前后的单词和句子联系起来。例如假定用 an 作为"动物"的标签。et 和 fd 分别作为"吃"和"食物"的标签,那么 Racter 所选择的句子形式就成为:

THE noun. anverb. 3p. etTHE

noun. fd.

这时 Racter 就要在自己的文件中查找一个名词,但是仅限于带有标识符 an 的名词。这样,它就会在从 AARDVARK(土豚)到 ZEBRA(斑马)的这些名词中去挑选。假定它选择了 MONKEY,接下来就要再选择一个带有标识符 et 的动词。这类动词中可以包括 EAT,MUNCH(大嚼)、NIBBLE(轻咬)等等。Racter 随机地选择了 CONSUME(吃光),并按照句子形式中的编码 3P 的要求将其变成第三人称过去时的形式。最后,Racter 寻找带有标识符 fd 的名词。并选择了 ANCHOVIES(鳀油)。其结果是得到这样的新句子:

THE MONKEY CONSUMED

THE ANCHOVIES

(猴子吃光了鳀油)

这个句子显然要比上一个句子更有意义。

Racter 的能力远远不限于在标识符的制约下搜索文件。它也完全有能力产生自己的句子形式。例如,假设动物和食物是刚才谈论的对象,那么 Racter 就会选择粗略的句子形式,并在这种句子形式内放上标识符。

事实上,Racter 在某处能产生自己的指令串并把它们嵌入到递



归流程中去。因为它始终必须遵守语法形式，所以产生的句子总是合乎语法的。由于使用了标识符，而且 Racter 保持着载有现在谈话中正用着的标识符的表，所以该程序可以提起谈话中讲到过的任何话题，至多只是改头换面而已。

上面的描述只是谈及了 Racter 全部操作的一些皮毛。我自己对该程序的理解比这多不了多少。但是我毫不怀疑，Racter 很快就会有许多仿效者，这种程序的一般原理也会进一步发展起来。我期待出现一本论述这问题的著作。

31. 词梯与通天塔

没有人知道火星上的文明是怎样消失的，也无人了解其消失的原因，但在奥林匹斯山脚下的一个窑洞内却保存有这个古代灿烂文化的一些可怜的遗迹。在窑洞尽头有一块碑，很像举世闻名的罗塞达碑，一张不知是什么东西制成的满是灰尘的织物盖在这碑上。揭开织物，便露出一个有点像辞典的东西：碑的左边是一串单词，右边与其相称的位置上又是一串单词。碑的底部刻着两句话。这两句话之间的关系非常奇特。只要反复地用辞典中的词来作代换（即将第一句中的出现于辞典左侧的词用在右侧的相应的词来代替），就可以把第一句变成第二句。

Martiansentencescontainno spaces（火星语的句子中不留间隔）。的确，由于古火星语具有这种灵活多变而又富于表达能力的特点，我们无须求助于任何人了；火星语的句子通常都可以找到许多方法将其划分为单词。不过，在特定的上下文中通常只有几种划分是有意义的。我们可以用本段的第一句来阐明上面所说的代换过程。如果tainnospace 出现于辞典的左边一栏，而 cealwisdom 是辞典右边一栏



与之相应的词,则可以用后面这个词来代替前者,这样就得到一个新的有意义的句子:

Martiansentencesconcealwisdom(火星语的句子中包含着智慧)

火星语的翻译问题只是整整一大类智力难题中的一个,这类智力难题要求将单词、句子甚至是整段文字转换成其他单词、句子与段落。这一领域在其发展初期的进展部分应归功于 Charles Dodgson 牧师(又名 Lewis Carroll)。他在牛津散步时想出了许多数学以及符号的游戏,所谓"词梯"(word ladder)的变换游戏便是其中之一。选定一种语言,词梯的变换过程便可从该语言的两个具体单词开始。第一个词称为源词,第二个词称为目标词。能够通过一次改变一个字母把源词变成目标词吗?

如果两个词的字母数相同,那么完成这一任务简直是轻而易举。但你能保证所有的过渡字符串也是单词吗?这个问题就是另外一回事了。不过文字游戏乐园中的老手驾驭这一问题也并不困难。事实上我们可以从"horse"(马)这一单词出发来说明这个变换过程。第一步可以把这个词变成"house"(房子),第二步就把它变成了"mouse"(耗子)。通过这种法术似乎可以把一匹马变成其名字有五个字母的任何东西。这是否可能呢?

为了从一般的角度来回答这个问题,有必要构造一个变换网络。我不喜欢哗众取宠的术语,因此给这网络另外取了个名字叫"词网"。完整的英语词网将是一个了不得的庞然大物,因为每个英语单词都将在这张网中占据一个单独的点。按照词梯的要求,如果有两个点所对应的单词恰好只有一个字符不同的话,就把这两点用直线连接起来。为了找出能从"horse"变出的单词,只需从"horse"这个词出发对词网进行搜索,把搜寻到的所有单词列成一张表。完整的搜索相当于抓住"horse"这一点把词网提起,从而把所有与"horse"相连的点都拉出来。这样得到的一大堆东西看起来大概像一张结得乱七八糟



的渔网。当然,大多数词不会跟着"horse"被提起来;比如,所有不是由五个字母构成的单词就仍然留在下面。

是否所有的由五个字母构成的单词都会被提起来呢?也就是说,是否能构造一个连接horse与其他任何一个五字母英语单词的词梯呢?虽然我必须说我并不知道这个问题的答案,但根据过去本专栏的问题所引起的读者的反应情况来看,我愿意断定有人会找到这一答案的。解决这一问题的必需条件是有一本计算机可读的辞典及一个在辞典中搜索有联系的词的程序。每当所有与"horse"相联系的词的集合中新增加一个词,程序就把这个词从存贮在存贮器中的辞典上删掉,然后再次通过整个集合,把集合中的每个词与它在不断缩小的辞典中所碰到的每个词相比较。一当在辞典中发现一个词与集合中某个词仅相差一个字母,就将这词加进集合中去,整个过程又重新开始。不同的辞典肯定将得出不同的结果,对吧?大概如此不过,对于相当完全的两本辞典,结果或许会是一样的。换句话说它们所产生的词网可能会非常厚,以致其拓扑结构实际上相同。

关于n字母单词是否都能互相连系起来的问题,对于n值较小的情况,基本上可通过人工来回答。所有的单字母单词肯定能连接在一个词网内。对于双字母单词,问题也能马上解决。根据我所用的Scrsbble辞典,双字母单词有:aa,ad,ae,ah,ai,am,an,ar,as,at,aw,ax,ay,ba,be,bi,bo,by,da,de,do,ef,eh,el,em,en,er,es,et,ex,fa,go,ha,he,hi,ho,id,if,in,is,it,jo,ka,la,li,lo,ma,me,mi,mu,my,na,no,nu,od,oe,of,oh,om,on,op,or,os,ow,ox,oy,pa,pe,pi,re,si,so,ta,ti,to,um,up,us,ut,we,wo,xi,xu,ya和ye。

不难看出所有这些单词在词网中是相互联系的。首先,第一个字母相同的所有双字母单词可以互相联系起来。这样就可以把双字母单词按其第一个字母分别归并起来。剩下就只是要注意到以"a"开头的那一组单词与其他四组以元音开头的单词是相互联系的,以

“b”开头的那一组单词与其他16组以铺音字母开头的单词是相联系的，而 a 组与 b 组本身也是相互联系起来的。

虽然仅有上面这两种n值较小的情况作基础，有人仍会大胆地猜测所有的n字母英语单词也是互相联系起来的。这一命题典型地表明了抽象文字游戏所引起的激动情绪。有计算机的读者可能会对另一些较小的n值（如n = 3）检验这一命题是否正确（见图1）。没有计算机的读者便只得满足于将“hate（恨）”变成“love（爱）”。更好的情况是可以尝试一下将“evil（邪恶）”变成“good（善良）”，从而获得对这一过程的道德与词逻辑方面的宝贵认识。

图1　三字母单词的词网

词梯游戏可以作为所谓的文本转换系统的一个简单例子。这样一个系统的两个主要构成部分是要转换的文本集合及转换规则。文本可以是单词、句子或段落，其实就是一些字符串，无论文本集合是如何选择的。转换规则就是把给定字符串的一部分用另一字符串来



代替的一系列步骤。在按规则作转换时通常要查阅辞典。

例如,在词梯中,字符串集合由给定辞典中的全部英文单词组成。指定任何一个要转换的词,转换规则即产生出另一个与它只相差一个字母的且也能在该辞典中找到的词。这个例子中的辞典就是一张简单的单词表。

大多数已出版的辞典中的词条都是由用其他单词"定义"的词构成的。这里就包含了另一类转换链——比词梯更加复杂的一类转移链。例如,想象一下从某一个英文单词出发,查出它的词义并将出现于它的定义中的所有单词列成一张表。然后再考察这些单词的词义,将其词义中出现的所有单词列成一张综合词表,并把这一过程反复进行下去。由于英语的单词数有限(我不能保证其他的语言也是如此),因此这一过程很快就显示出一种可以说是"循环"的性质,即某些单词开始反复出现。它们在某种程度上是自己定义自己。可以将这类词称为"原始单词"。它们的出现并不会由原始的欢呼声来通报,而是带几分敬畏静静地重复。这类词大概构成了英语的不能定义的观念基础。

这一工作是由 Robert Amsler 进行的。他所使用的辞典已作了一些修改,使得每个词的定义都简化成一组类属词,这种类属词可以用来识别该词所归属的广泛的各类事物。类属词附有"种差",即用于把某一给定词从类属的其他成员中区别出来的词。借助一个与前面介绍过的程序相类似的程序,Amsler 检索了他的专用辞典并发现了基本上是语言哲学家们预言他将发现的东西:原始词。所有的原始词在其他原始词的最终定义中都曾用到过。原始词的例子包括 food(食物)、person(人)、thing(事)、instrument(仪器)及 group(群、组)等。原始词占据着一种所谓"缠结结构"(这种结构有点像不同的树的树枝长到一起的森林)的最外面的分支。它们位于普遍性逐渐增大的这一方向的"上侧"。



Amsler的基本想法是,辞典的结构比人们一般认为的要复杂,辞典所能揭示的有关语言的信息也比人们一般认为的要多。如果这一比较简单的种属-类差方案没有产生出一些异乎寻常的东西,那将是非常奇怪的。例如,barbecue原来是"在火上烧烤的动物"。这样barbecue就发现它自己在一张表上位于aardvark(土豚)到Zebra(斑马)之间的某个地方。

涉及到《新版Collins同义词辞典》。读者使用同义词辞典时通常都是为了寻找一个单词的同义词或者寻找他所想到的属于某一概念类别的词。可以想象每一个词都藏在由与其意义实际上相同的若干个词组成的一小朵"词云"中。但这些词云并非全都是相互隔得很开的,有些词云与其他词云相互间有重叠部分。其结果是,虽然某些成对的词本身并不是互为同义词,但它们却可能有共同的同义词。由此可以弄出许多令人哭笑不得的结果来。我们可以看看下面几个由Hardin发现的转换链,其中每一个词都位于其前面一个词的"同义云"的范围内:

acceptable(可接受的)→so→so(平常的,一般的)→ordinary(平常的,一般的)→inferior(劣等的、次的)→rotten(糟糕的)→unacceptable(不可接受的)。

reliable(可靠的)→steadfast(不变的、坚定的)→obstinate(顽固的、难治的)→wayward(任性的、反复无常的)→unpredictable(不可预测的)→unreliable(不可靠的)。

借助计算机已经生成了好几千个这种转换链。每个转换链都是使一个词逐步地转变成其反义词。所生成的转换链中,绝大多数的长度都不超过上面这两例。此工作提醒我们注意这一点,即人类语言是变化多端的,单词的含义与上下文有很大的关系,而且每个词都有其模棱两可之处。这一点既是幸事,也可能成祸端。试看下面这个例子;律师就被告的性格问题反复盘问证人:



“您在证词中已经申明 Watson 是一个可靠(reliablc)的人。您是否也会说他是一个‘坚定’(steadfast)的人呢?”

“当会然。”

“说实在的,您是否会说有时被告在其坚定之中带有了一点顽固(obstinate)的味道?”

“唔……我想是这样吧。”

“那么,有时他的这种顽固不化肯定差不多等于某种形式的任性(way wardness)了,是吧?”

“这个我可不能肯定了。”

“不能肯定? Finch 大夫,您瞧,顽固跟任性实际上是一回事吧。”

“我想是的。”

“一个任性的人就是一个不可预测(unpredictable)的人,简而言之,也就是一个不可靠(unreliable)的人。因此,您刚才已经作出了与您早些时候的证词相矛盾的论断。”

图 2 列出了其他几个这种由一个词变到其反义词的序列的例子。使用“新版 Collins 同义词辞典”的方法并不像上面叙述的那样简单。该辞典是按主要词编排的。当使用该辞典的读者想要找出一个代表某一给定内容的新词时,首先要查到其意义与他所考虑的内容最为接近的主要词。在主要词之下列出的是各有关词。这些有关词按词义排列。例如,在“express”(表达、显示、快递、快车)这一主要词下可以找到与言辞及速度有关的词义。主要问题是以一种合乎逻辑的、前后一致的方式把各词义通过主要词联系起来。

因此,人们设计了一种算法,该算法可以系统地检查所有的相关主要词对,并计算每个主要词可以取的各种词义间的联系的程度。然后,把具有最大程度的联系的词义加进由该算法所产生的一种词网中。使用这一方法,词义间的区别一般就保持在相当清楚的程度上,但有时仍会变得比较模糊。在这个词网完成时,第二个算法就投



afraid – apprehensive – expectant – hopeful – confident – **unafraid**
classified – secret – mysterious – unidentified – **unclassified**
dress – groom – tidy – clear – wipe – take off – divest – **undress**
even – level – uniform – regular – periodic – spasmodic – **uneven**
perfect – pure – unvarnished – unfinished – rough – **imperfect**
potent – dynamic – animated – excited – nervous – weak – **impotent**
rational – clever – witty – funny – silly – brainless – **irrational**
safe – undamaged – mint – untarnished – polished – slippery – **unsafe**
social – friendly – peaceable – inoffensive – retiring – **antisocial**
valid – convincing – plausible – specious – unsound – **invalid**

图 2　Ron Hardin 的通天塔

入使用,该算法的作用是找出任意两个词义间的最短路径。这样就很容易规定出各种不同的词对,从而在圣经上的通天塔因语言上的混乱未能完成之后创造出了他自己的通天塔。通天塔是一种微型的巨作,它对语言学家及律师们有特殊的吸引力。

现在剩下要做的事就是回到前面所说的火星语翻译的问题上。给定一本允许用于替换的辞典与两个词(无论多长),为什么计算机不可能从原则上确定第二个词是否可以通过辞典进行一系列替换从第一个词得出呢?这个问题称为 Thue 词问题,因挪威数学家 Axel Thue 的名字而得名。

计算领域之外的许多人(甚至从事该领域内的工作的某些人)都没有意识到有些事情是计算机不能干的。最早提出计算机有其内在局限性的人之一是英国的 Alan M. Turing,他是计算机科学的奠基者。Turing 设想了一类简单的抽象计算机,称为 Turing 机(但这并不是他取的名字)。这种机器原则上能够执行现有的或设想的任何一台计算机所能作的任何计算。Turing 证明了这一事实:对一台 Turing 机来说,给定了另一台 Turing 机及其输入,它不可能确定这第二台 Turing 机会不会停下。更具体地说就是,不可能编写出这样



一个程序，对于给定的第二个程序及作为其输入的数据，它能确定这第二个程序是否会停止处理其数据。

Turing 的定理至少有一个实际用途。某教育机构曾有一位系统论专家，他断定一台计算机主机在执行新手所编写的计算机科学程序上花的时间太多。这种拖沓现象的主要原因似乎是许多程序中都包含有无意中造成的无穷循环。如果在执行程序之前，能够用一个可检出无穷循环的程序来检查这些程序，那就可能节省许多时间。不妙的是，正如一位 Turing 的学生很快就告知这位专家的那样，能检出无穷循环的程序是不可能编写出来的。

既然已知这个所谓的"停机问题"是不能解决的，我们就比较容易向一般人证明火星语的翻译问题也是不能解决的。通过证明停机问题可以转化为火星语的翻译问题可将论证过程继续进行下去。

这一转化从概念上说是比较容易实现的。计算机编码为一长串符号。这个符号串基本上代表了计算机存贮器与工作寄存器的初始内容。然后构造出第二个符号串。它表示计算机处于某一停机状态；此时所有预定要做完的计算都已完成了。要进行停机检验的程序被翻译成一组通过替换将第一个符号串转换为第二个符号串的转换规则。随着计算机的操作按我们所希望那样向着完成数据处理的方向不断进行，每一个中间符号串都表示这一操作过程的一个步骤。转换规则遵循程序的规定，利用替换过程将这台虚计算机从一个状态变换到另一个状态。

例如，如果一个寄存器的现行内容要存贮在某一存贮器地址中，转换规则就将那个存贮地址的新内容代入现行符号串中代表所考虑的存贮器地址的那部分。当且仅当所论的计算机最终将停止对提供给它的数据的处理操作时，这一转换步骤才能够从第一个字符串中获得第二个符号串。既然 Turing 定理已经表明这计算机永远不会停机，因此这一转换是不能实现的。

32. 程序创作的散文及诗集

正如一位著名的研究无意义语言的牛津学者所说，在文学创作中，语义问题比句法问题更重要。优秀的文学作品总是包含了作者深刻的思想内涵。计算机还不能思想，因此也不可能寓意于一部作品。然而，正如许多现代程序所表明的那样，计算机肯定能够处理作品的语言。但那是艺术吗？这得由读者来评说。

看看程序 MARK V.SHANEY 的作品吧。程序 MARK V.SHANEY 并不是一个完全靠自己从头开始进行创作的程序。它在进行创作时必须先阅读一篇别人的文章并对其作一番思考，然后才能产生自己的作品——一篇不连贯的散乱的评论。作为例子，我引用了 MARK V.SHANEY 阅读了一本初级化学课本后作出的评述：

“擦洗黑板。观察黑板的干燥过程。水进入空气。当水进入空气时，水就蒸发了。把一条湿布系于一个固体或一瓶液体的一端。看看周围。固体是什么？那些进入云层的东西是什么？当云层移动时，空气就使得云层中的小水滴上下颠簸。有时从云上掉下的水进

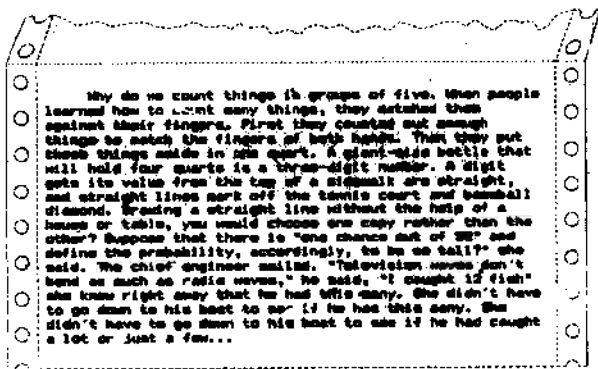


入了绿色植物的叶片。绿色植物有相当多的孔,几乎就像海绵。然而,空心导管壁和细胞壁都是很结实的。当物体燃烧时就发生了化学变化……"

当 MARK V.SHANEY 阅读并领会了一本初等数学书后,它作出的评论也没有多大不同:

"为什么我们五个五个一组地计数呢?在人们学习数许多东西时,往往扳着自己的手指头来点数。首先他们点出恰好与自己双手的全部手指头一样多的东西。然后把这样多的东西归为一夸脱。一个能容纳 4 夸脱的大容量瓶是一个三位数……"

从这里开始,程序的输出越来越令人莫名其妙,读者自己可以从图 1 中看出这一点。



Why do we count things in groups of five. When people learned how to count many things, they matched them against their fingers. First they counted out enough things to match the fingers of both hands. Then they put these things aside in one quart. A giant-size bottle that will hold four quarts is a three-digit number. A digit gets its value from the top of a sidewalk are straight, and straight lines mark off the tennis court and baseball diamond. Drawing a straight line without the help of a house or table, you would choose one copy rather than the other? Suppose that there is "one chance out of 25" and define the probability, accordingly, to be so tall?" she said. The chief engineer smiled. "Television waves don't bend as much as radio waves," he said. "I caught 12 fish" she knew right away that he had this many. She didn't have to go down to his boat to see if he has this many. She didn't have to go down to his boat to see if he had caught a lot or just a few...

图 1　MARK V.SHANEY 的数学评论

虽然 MARK V.SHANEY 的作品明显缺乏有条理的内容,但是语气肯定是存在的。给人的整个印象就像一位漫不经心的学生熬夜读书后大脑中所留下的零乱知识一样。

MARK V.SHANEY 是怎样产生出这些引人注目的作品的呢?答案相当简单。这个程序的名字实际上是一个双关语,它隐含了



"Markov chain"(马尔可夫链)这一意义,从而为我们提供了一条线索。概括地说,马尔可夫链是根据一张概率表产生的符号序列。在运行程序 MARK V.SHANEY 时所要用到的那种概率表中,每一行都与一对符号相对应。每一行中的条目都是一些单个符号,每个符号都有一个与其相关的概率值。一个算法从只有两个符号的初始链开始,反复经过四个简单的步骤进行循环,便可产生出一连串的符号。这四个步骤是:

1. 找出当前链的最后两个符号。
2. 查看概率表上与这一对符号相应的一行的各概率值。
3. 根据其概率从该行中选出一个符号。
4. 把选出的符号加到该链的末端。

例如,一个有 A、B、C、D 四个符号的马尔可夫链表的最初几行可以是:

AB:B(0.1)　C(0.1)　D(0.8)
AC:A(0.1)　B(0.2)　C(0.1)　D(0.6)
AD:B(0.4)　D(0.6)
BA:B(0.3)　C(0.4)　D(0.3)
BB:A(0.5)　C(0.5)

如果给定符号对 AB 作为初始链,则算法可以有 10% 的机会选择 B、10% 的机会选择 C、80% 的机会选择 D 作为下一个符号。算法怎样根据这些概率来选取符号呢?它把从 0 到 1 这一段区间划分成若干数字段,各段的长度与各个符号的概率相对应。然后算法在 0 与 1 之间选择一个随机数并决定该数处于那一段。

对于上表中的 AB 行,与 B、C、D 的概率相对应的数字段分别是从 0 到 0.1,从 0.1 到 0.2 和从 0.2 到 1。假定计算机的随机数生成程序产生的随机数是 0.0172。因为该数落在第一个数字段,因此 B 被选出作为该链的下一个符号,该链将由符号 ABB 组成。算法下一

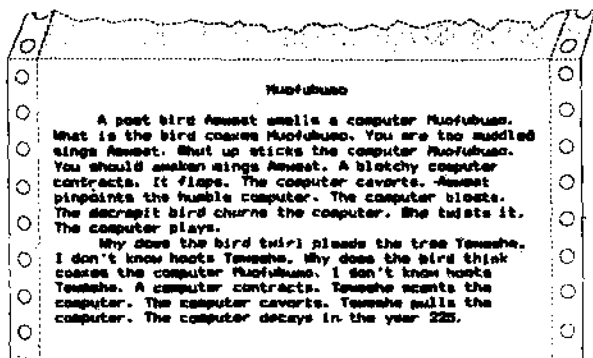


步将查看概率表上与符号对 BB 相对应的那一行，以便为该链选取第四个符号。此时随机数生成程序又产生一个随机数，如果这个数小于或等于 0.5，A 将被选取，否则算法将选取 C。由于算法对符号的选择是完全随机的，因此，即使算法由同样的初始符号对开始重新运行，它也非常可能产生一个完全不同的符号链。实际上，类似的这样一种算法在 20 世纪 40 年代已由 Claude E. Shannon 用来分析人类语言的信息容量。他通过扫描普通的文本并数出每一个字符紧接着各对字符(空格也算一种字符)后出现的次数，作出了一个算法概率表。一旦给定文本中的各个字符出现的频率已知，那么把这些频率转换成概率就很容易了。用这种方式产生的马尔可夫字符链具有与原始文本相似的统计性质，尽管它们很少包含有意义的词。那么程序 MARK V. SHANEY 是怎样应用马尔可夫链来产生可以理解的英语单词的呢?

奥妙在于，MARK V.SHANEY 在把马尔可夫链的 Shannon 算法用于构造符号串时，不是用单个字符而是用整个单词作为符号串中的一个符号。当 MARK V.SHANEY 扫描一份文本时，它就建立了文本中的每个单词紧跟在各个单词对之后出现的频率表，然后程序就以词频为基础按照统计方法生成自己的文章。

这一程序的关键特征是将与某一单词相邻的任何标点符号看作该单词的一部分。这一特征使程序能够形成完整的句子，其中近乎一半甚至具有语法结构。我一想到程序在扫描本文后可能产生出的结果便战栗不已!

确实，其他有些人已经为 MARK V.SHANEY 的作品而震惊。一些人愤怒，一些人嘲笑。几年以前，当这个程序上网时，那些使用一种被称作“net.singles”的电子公告栏的无辜的计算机用户成了 MARK V.SHANEY 的恶作剧的牺牲品。这一电子公告栏为全国各地的男女科学家、工程师、程序员以及研究生提供了一个发表他们对



Muofubuso

A poet bird Amuset smells a computer Muofubuso. What is the bird coaxes Muofubuso. You are too muddled sings Amuset. Shut up sticks the computer Muofubuso. You should awaken sings Amuset. A blotchy computer contracts. It flaps. The computer cavorts. Amuset pinpoints the humble computer. The computer bloats. The decrepit bird churns the computer. She twists it. The computer plays.

Why does the bird twirl pleads the tree Towashe. I don't know hoots Towashe. Why does the bird think coaxes the computer Muofubuso. I don't know hoots Towashe. A computer contracts. Towashe scents the computer. The computer cavorts. Towashe pulls the computer. The computer decays in the year 225.

图 2　Firner 的程序 MELL 讲的故事

诸如年龄、性格以及个人关系等形形色色的问题的看法的地方。为什么不让 MARK V.SHANEY 读读这些看法并就这些问题阐述它自己的想法呢？下面是 MARK V.SHANEY 的一些更富思想性的评论：

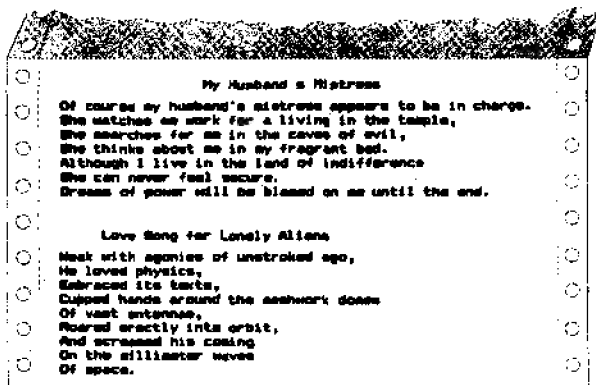
“当我在某个专门的军事基地遇见某个人时，我要他们削减武器。在几周前的一次会议上，我度过了一个颇为刺激的有趣的夜晚：我可不敢和他们当真！这使我又成了一个敢作敢为的莽汉。我喜欢友好热烈的争论，可能比其他的人更喜欢……”

“我打算引入一个新话题：有意见吗？还有什么人打算就异族夫妇，或者叫异族通婚，进而或者说是种族歧视以及眼影等方面的经历发表见解吗？这可能是本性，虽然它值得一读，更不用说判断另一个人读了没有？天哪！”

这位新加入 net.singies 网络的通信者的见解引来了各种不同的评论。电子公告栏的忠实读者认为受了讽刺。他们大为愤怒，要求把 MARK V.SHANEY 从网络中驱逐出去，不准它胡说八道。而另一些人则近乎羡慕地询问此程序是否是一个正在人类对话环境中进



行试验的秘密人工智能项目。有人甚至认为 MARK V.SHANEY 是一个真正的人,一个遭受精神折磨的极度渴望寻找有同样想法的伙伴的精神分裂症患者。



My Husband s Mistress

Of course my husband's mistress appears to be in charge.
She watches me work for a living in the temple,
She searches for me in the caves of evil,
She thinks about me in my fragrant bed.
Although I live in the land of indifference
She can never feel secure.
Dreams of power will be blamed on me until the end.

Love Song for Lonely Aliens

Meek with agonies of unstroked ego,
He loved physics,
Embraced its tests,
Cupped hands around the meshwork domes
Of vast antennae,
Roared erectly into orbit,
And screamed his coming
On the millimeter waves
Of space.

图 3 POETRY GENERATOR(左)和 THUNDER THOUGHT(右)向文人学士们提供挑战

如果用计算机作散文的目的是要哄人相信其作品出自神智健全者之手的话,那么 MARK V.SHANEY 可能没有实现这个目的。新泽西州皮斯卡塔威的 Bonnie V. Firner 研究出的程序 MELL 更为接近这一目标。MELL 用一种特殊的思维方式写了一些离奇的科幻故事。

“一位武士在干旱的豆地里皱着眉。他喜欢自己在旱地里。又把自己铸造在旱地里。他怒视着旱地里的武士 Dugaki。他向她怒吼。他全神贯注地坐在旁边。他捕获了她。”

“我是强大的,Oban 说。他把她窒息在旱地里,又用绿松石猛击她……”

“Oban 杀死了 Dugaki,他是用绿松石打死他的。他在旱地的豆



地里怒视着它。又在旱地里呼叫着它。他在旱地里对着它怒吼,并贪婪地坐在其旁。他捕获了它,又用绿松石猛击它。他在雷电下需要它,他的身体在雷电下燃烧起来……"

程序 MELL 由大约 1 500 行 BASIC 代码组成。与 MARK V.SHANEY 相比,Firner 的程序即使在观念水平上也是相当复杂的。程序的主循环用 16 个随机产生的描述符表述两个人物。描述符的值决定诸如身材、教养、职业、年龄、健康、气味、参与程度(从冷漠到狂热)以及魅力等。

确定人物的姓名及特征后,程序的主循环就通过检查不同的描述符的值来决定人物的动机。如果一个人物的描述符中有一个的值较低,MELL 就根据这一事实确定此人物与其他人物的相互作用。如果所有人物的描述符的值都不低,那么 MELL 就根据他(或她)的职业来决定人物间相互作用的性质。主循环每运行一次,便把描述人物特征的词填入预先确定的语法位置上,从而产生描述人物的几个句子。这样生成的句子中还包括了 Firner 所谓的"背景"词。

例如,在上面的故事中,背景词"旱灾"便交待了场景。因此该词也出现在其他许多句子中。程序像这样产生出一段描写其中一个人物的活动的段落中,整个造句过程又全部从头开始。在两次循环之间,程序可以改变其中一个人物的特性,这样就避免了呆板,保持了故事的生动。

用计算机作诗比计算机作散文更容易吗?人们怀疑这一点,因为诗歌中所含的内容远比散文深邃。在我所要讨论的三个作诗程序中,只有加利福尼亚州米申山的 Rosemary West 编写的程序 POETRY GENERATOR 是全自动的,其他两套程序则需在人工干预下才能完成其作品。West 是这样描述该程序的:

"我的方法是提供一个包含大量单词和词组的词汇表,这些词可以随机选取并可根据一套语法规则组合起来。例如,请看下面这首



诗:“树儿垂下枯萎的枝头/埋在黑色的冰地里/就像三只灰色的母鹅/滑向近旁的雪堤。”

“这首诗的每一行都可以分割成几部分……树枝是名词短语;‘垂下’是动词,‘枯萎的枝头’是动词的宾语。将各部分归类后,我可以为每一部分提供100个到400个替换词,计算机从这些替换词中随机选取一个。例如,采用同样的诗文结构,我可以得到这样的诗:一位妇人藏了五只灰色的小猫/在破旧的汽车里/此时那位哀伤的乡农/唤起你痛苦的回忆。”

程序POETY GENERATOR用来填词的诗的结构可以有许多种,从而使诗的句法及诗的内容富于多样性。West曾将POETRY GENERATOR的输出建立在她自己写的诗文结构的基础之上,这些诗中有几首已出版。

缅因州贝尔法斯特的Thomao A. Easton认为产生计算机诗歌的最好方法必定要利用人机之间的“共生”现象。他编了一套称为THUNDER THOUGHT的半自动程序,该程序可以为诗歌提供构思。这里我仍然引用作者的一段话:

“我认为有创造性的头脑应包括两方面的要素:一方面是突发思维;另一方面是批判思维。前者对于内心涌动的词汇、构思、意象以及一些边缘素材进行思维聚焦,产生出某些随机的组合(这就是突发思维能令我们惊讶的原因)。后者则对突发思维产生的大量素材进行筛选,将其中许多作为无用的垃圾扔掉,然后对剩下的素材进行编辑,剪裁和加工,以形成真正的诗歌。依赖内部存储的名词、动词、形容词和副词表,THUNDER THOUGHT就大致可以像West的“POETRY GENERATOR”一样运行,将词汇填入句子结构中以形成Easton所谓的“原始诗歌”,然后再由人工加以进一步的润色。

“这样得出的中间结果是不合语法的、无意义的、荒谬的东西……但也不总是如此。在众多的无意义的诗行中总有那么几行引人

注意。这几行是有条理的(或者似乎有条理的)。它们促使人们去对它们作点加工。发现两行有意义的诗句,人们就会禁不住想作出第三句。有意义的诗句总是激发人们去想出能与它们相配的更多的诗句。稍加编辑、发挥和提炼,一首诗便出笼了。"

Easton 用这种方法写了 110 首诗,其中 32 首已经发表,有的甚至发表在文学杂志上。他声称,这样高的成功率令许多真正的诗人妒忌。

最后来讲一下程序 ORPHEUS。ORPHEUS 这套程序设计了严格的诗体(从三行诗到十四行诗都有),人类作家可将他们自己选定的词填入这些结构中。ORPHEUS 基本上是一个字处理程序,它按给定的诗歌格式安排诗行。这一程序允许人们根据其构想在各行中填词,然后借助诗韵词典使每行的末尾一词符合诗韵。例如,将该程序置于十四行诗的格式上运行,程序就可写出如下的两行诗(模仿莎士比亚的第 130 首十四行诗)。

我的苹果机的显示屏一点不像太阳;克雷机却是运算难题的快手:

因为第一行结尾词是"Sun",程序通过查阅诗韵词典显示出许多与 Sun 同韵的词:bun、done、fun、gun 等。浏览一下这些词,人们的目光也许会落到"fun"这个词上还有比苹果机更有趣的计算机吗?

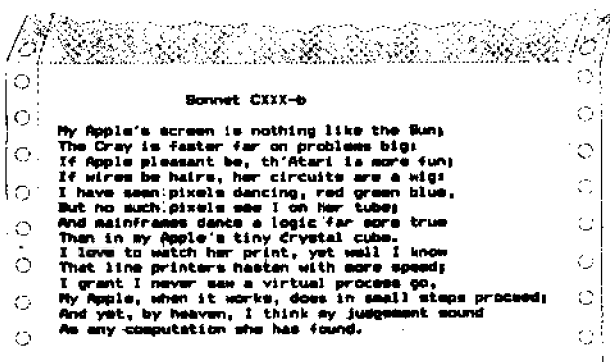
无独有偶,程序灵机一动想起了阿塔里机,于是把它填入下一行:

苹果机逗人爱,阿塔里机却更迷人。

第一首四行诗选取一个与 big 同韵的词就可完成:

如果电线是头发,它的线路就是假发。

这首十四行诗的其余部分见图 4。



Sonnet CXXX-b

My Apple's screen is nothing like the Sun;
The Cray is faster far on problems big;
If Apple pleasant be, th'Atari is more fun;
If wires be hairs, her circuits are a wig;
I have seen pixels dancing, red green blue,
But no such pixels see I on her tube;
And mainframes dance a logic far more true
Than in my Apple's tiny crystal cube.
I love to watch her print, yet well I know
That line printers hasten with more speed;
I grant I never saw a virtual process go,
My Apple, when it works, does in small steps proceed;
And yet, by heaven, I think my judgement sound
As any computation she has found.

图 4 一位不知名的作者与 ORPHEUS 共同构思创作的十四行诗

33. 形形色色的细胞自动机

水的分子居然"知道"如何构造雪花,使其具有复杂的对称性,这的确令人赞叹不已。这些分子自己能够装配起来,既没有建筑师的指导,也没有结晶形式的模板。大尺度的花样完全是依靠许多相同单元的短距离相互作用而产生的。每一个分子仅仅受到最近邻域的影响,然而在大约多达 10^{20} 个分子的结构中却能够保持某种排列的一致性。

要理解这种过程,一种方法就是设想分子点阵的每一点都由一台初级计算机来控制。随着晶体的生长,每台计算机都要巡视周围的点,根据它所发现的周围的点的情况并依据某种固定的规则来决定它自己的这个点究竟是要占据还是空着。在所有点上都按照同样的规则进行同样的计算。

这种雪花生长的计算机模型就是细胞自动机:许多相同的细胞(即格点)以均匀的方式排列,每一个细胞只有几种可能的状态,并且只和相邻的几个细胞有相互作用。整个系统的组成部分,即细胞和计算某个细胞的下一个状态的规则都可以非常简单,但是它们仍会



导致非常复杂的发展过程。

细胞自动机的思想几乎与数字电子计算机的历史一样久。20世纪50年代初,冯·诺埃曼(在 Stanislaw Ulam 的大力协助下)便首先对此进行了研究。他最初的目标是要设计一种能像生物体那样自我繁殖的简单系统。John Horton Conway 在1970年发明的著名细胞自动机“生命游戏”,正像这个名称所表明的那样,具有生物学的含义。细胞的产生、成活或者死亡,都取决于局部的细胞密度。

关于细胞自动机的更近的研究工作,重点有所转移。局部相互作用细胞点阵被看成是对于物理系统可能非常有用的模型,这些系统包括雪花、铁磁体、星系等。它们也可以应用于计算机科学,既包括实际方面(由许多计算机联结而成的网络应该如何组织?)也包括理论方面(计算机能力的最终极限是什么?)。也许最使人们感兴趣的是可以把细胞自动机看成是一个“数字宇宙”,即使完全不考虑把它作为实际世界的模型,其本身就是值得人们去探索的。

细胞自动机具有四个特点。第一个特点是细胞点阵的几何学。对于雪花生长的模型来讲,二维平面上的六角形点阵是合适的,但在大多数情况下可以选择交叉直线构成的点阵,它由同样的正方形组成。三维或更高维的点阵也很容易构造,但是很难用直观的图像来显示。最近的一些惊人发现却是在更简单的一维点阵中得到的,这种点阵仅由一条线上的细胞组成。

在一个给定的点阵中必须规定每个细胞在计算自己的下一个状态时所考察的邻域。在二维的直线型点阵中,有两种邻域最值得注意。冯·诺埃曼限定每个细胞只注意它的4个最近的邻居,即东、南、西、北的4个邻居。它们称为冯·诺埃曼邻域。以 Edward Moore 命名的穆尔邻域包括上述4个邻居和另外4个对角相邻的邻居。显然,邻域是相互重叠的,某一个细胞可以同时在几个相邻细胞的邻域之中。有时也可以把进行计算的中心细胞看成是在它自己的邻域之

内。

在细胞自动机中要考虑的第三个因素是每个细胞可能出现的状态数目。冯·诺埃曼发现了一个自我复制模式，它的细胞具有 29 种可能的状态，但是大部分细胞自动机则要简单得多。即使在二元自动机（即每个细胞只能有两种状态的自动机）中，它的变化范围也是非常大的。这类状态可以表现为 1 或 0、真或假、开或关、生或死。

细胞自动机世界的多样性的主要根源在于，根据一个细胞的邻域的当前状态来确定该细胞的未来状态的可能规则是非常多的。如果每个细胞的状态有 k 种，邻域中包含 n 个细胞，那么可能的规则有 k^{kn}条。因此，对于具有冯·诺埃曼邻域（这里的 n 为 4）的二元自动机来讲，就可能有 65 000 多条规则。而在穆尔邻域（n 为 8）中则为 10^{77}。归根结蒂，迄今只考察了这些规则中微不足道的一部分。

在具有穆尔邻域的直线点阵中的两种状态细胞所玩的生命游戏还有一条规定是，中央细胞是不容忽视的。也就是说，在系统发展的每一步，每个细胞都要考察 8 个相邻细胞的状态，也要考察自己的状态。按照 Conway 所确定的规则，如果中心细胞是活的，而且在它的相邻 8 个细胞中有 2 或 3 个细胞是活的，那么它在下一代中仍然是活的。如果在邻域中有 3 个细胞是活的，那么不管中心细胞目前的状态如何，它在下一代都是活的。在所有其他情况下，中心细胞要么死去，要么保持死亡的状态（见彩图 9）。

生命游戏的妙处在于它的不可预测性。有些模式会消形灭迹；更多的模式会转变成为一种稳定的构形或者以几代的周期循环。但是这几年中发现了一些更加有趣的初始状态，例如“滑翔导弹枪”，它射出一连串无休止的“导弹”来。关于生命的其他次要方面的探索仍在继续进行。

在可能的大量转移规则中，有许多实质上是很乏味的。例如有条规则规定，一个细胞当且仅当它左侧的细胞是“开”时，它才是“开”



的。这种发展过程就很容易预测：任何一种初始的模式都会保持下去，只是每一步都向右移动一个细胞。有一类规则叫做计数规则或者总体规则，它似乎已经包括了已经观察到的细胞自动机的所有变种。根据这类规则，细胞的新状态仅仅取决于处于某一给定状态的相邻细胞的数目，而与它们的位置无关。

有一种最简单的计数规则就是奇偶性规则。如果一个细胞有奇数个相邻细胞是1，就给它赋予1，否则就赋予0。

还有一类计数规则是"投票"规则。只要邻域中具有值1的细胞数目超过一定的阈值，那么中心细胞就为1。Vichniac在递交洛斯·阿拉莫斯会议的一份报告中指出，这类规则可以产生渗透和成核的模型，这是固态物理学和其他领域中的两种重要现象。渗透这一术语指的是穿越某种空间的连续路径的形成。例如当把一种金属扩散入一种绝缘的基体中时，此复合体的导电性就取决于形成一条金属原子的连续链的几率。同样，一种传染病的传播只有通过一连串容易受感染的个人进行，而不能有中断。成核则引发晶体的生长、液体的沸腾以及类似的事件。

有一种引起渗透的转移规则是，在包括中心细胞在内的冯·诺埃曼邻域中，当且仅当5个细胞中至少有3个为1时，中心细胞也为1。渗透的发生对于原来1的密度是非常敏感的。如果密度小于0.5，在发展过程中就不可能形成连续穿越点阵的1的链。当密度等于或者大于0.5时，链就出现了，但是并非点阵中的所有点都为1；在最后的稳定态中仍然会有为0的孤岛。当规则变为只需要5个细胞中有2个为1时，可以观察到成核，此时固体点阵中充满了1。其临界密度为0.0822。

Ising模型是物理学的概念性工具，从表面上看来，它和细胞自动机非常相像。这个模型是直线型点阵，每一点只可能有2种状态，并且只和4个最近的邻点相作用。该模型常常用来描述铁磁体物

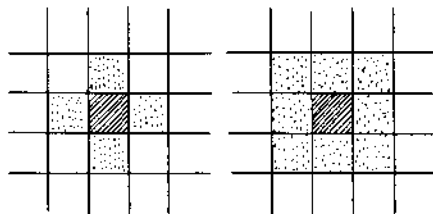


图1　冯·诺埃曼邻域和穆尔邻域

质,每一点代表一个原子自旋,它的指向要么向上,要么向下。低于临界温度(居里点)时,自旋倾向于定向排列,于是物质就被磁化了,但是在更高的温度时,它们或多或少是无规则分布的。

由全版面程序产生的一种 Ising 模型的细胞点阵很自然地适合于细胞自动机的研究,但其转移规则是随机性的,以模拟温度的作用。我观察到一种有趣的现象:在低温下,自旋并不是都取同一方向,而是取一种上下方向交替的方格结构。而每一步都是所有的自旋都倒转。在铁磁体中,方格模式是最高能量的结构,因而是不稳定的;这是一种具有反铁磁体特性的模式。

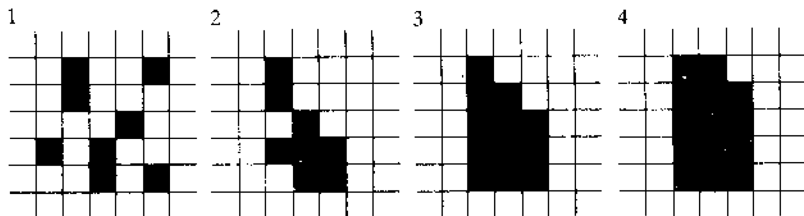


图2　服从五中取二投票规则的细胞自动机的发展过程

Vichniac 早就发现了这个问题并加以解释。在 Ising 模型的标准实现过程中,每次重复时只容许有一个自旋改变自己的方向。由此可见,当某一点巡视自己的邻居时,它所检查的自旋中有些是"老



邻居”，有些是“新邻居”。在这种情况下不可能产生振荡的反铁磁体。只有当所有的自旋都同时重新计算时，才有利于高能反铁磁体的产生。有一些办法可以避免这类“反馈灾难”，但是更有意义的教训是告诉我们，在 Ising 模型与细胞自动机之间的那种直觉对应关系是容易使人发生误解的。

Vichniac 和麻省理工学院小组的其他成员指出，细胞自动机与其他物理学模型有着根本的不同。长期以来，构造自然界数学模型最普遍的方法就是采用微分方程，它可以描述某个作为位置和时间的函数的量的变化。例如麦克斯韦方程就给出了电磁场随位置和时刻而变的变化情况。在这类方程中的所有量都是连续的：它们是平滑变化的。相反，细胞自动机完全是一种离散的系统。空间并不是连续的，而是细胞的点阵；时间也分割成离散的时段。场的量值可在一连续的范围上变化，而细胞自动机的细胞却只能有有限的几种状态。

当然，实际的空间和时间以及许许多物理变量都被看成是连续的，而不是离散的(至少在一般考虑的范围内是这样)。但是，并不能因此就得出结论，微分方程作为自然界的模型就一定是优越的。在许多情况下，重要的并不是某一变量的精确数值，而仅仅是该变量的总的范围。例如在确定正在生长的雪花中的某一特定点究竟是冰或是水蒸气时就是这样。

在数字计算机上细胞自动机使得这种离散性明确无误，并且可以精确计算它们随时间的演变；这时不再需要近似的方法。而且这样一来，就可以更加有效地利用数字计算机的资源。

有一种最简单的细胞自动机程序仅仅相当于用手在绘图纸上完成的过程。首先是确定一组细胞的点阵，每个细胞用计算机中的一个存贮单元来表示。每一步该程序都要轮流到每一个细胞考察它的邻域并计算细胞在下一状态的合适值。这种计算本身是很容易的，

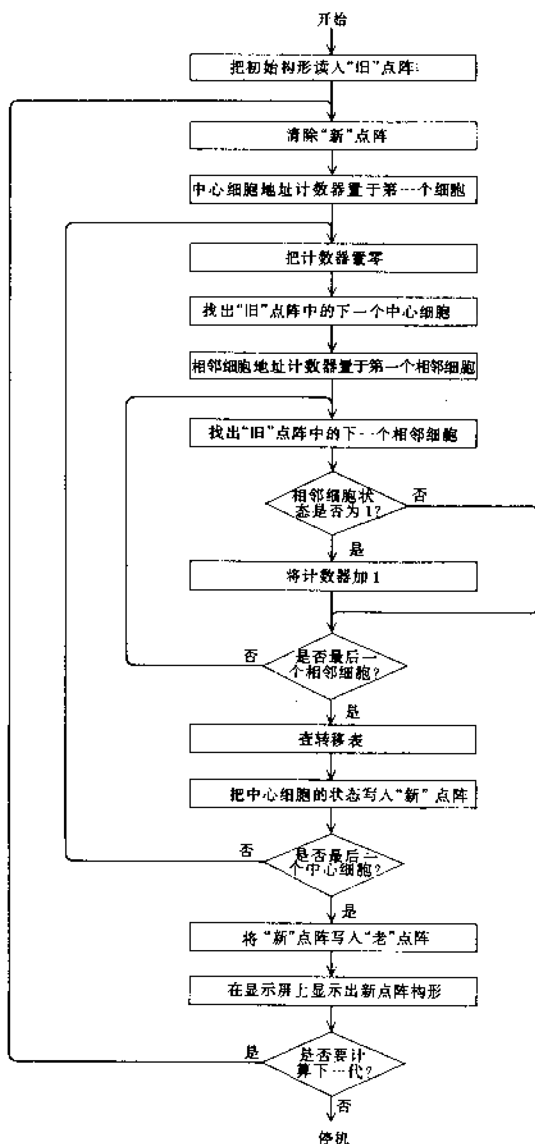


图3 根据"计数"或"整体"转移规则的细胞自动机算法



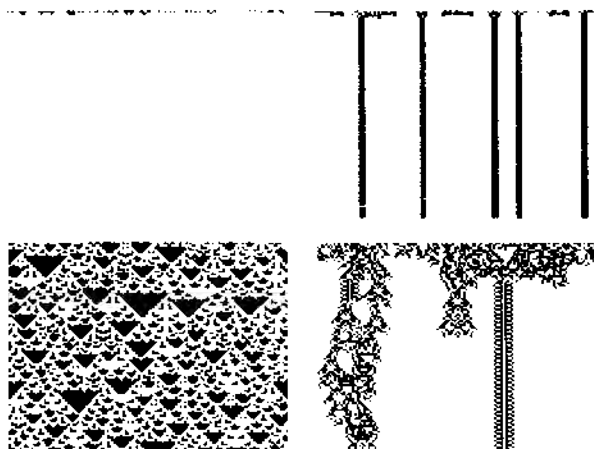


图4　4类一维整体规则

只要查阅一张表上的值就可以了。如果只考虑计数规则，则这张表相应于处于“开”状态的细胞的每一个可能的数目只需有一项就可以了。

如果要考虑其他的规则，那么这张表就会变得非常复杂。

当你编写这样的程序时一定要记住几点不易掌握之处。最重要的是在一个细胞还没有被所有其他相邻细胞都考察之前，不能改变该细胞的内容。要满足这一条件的最容易的方法就是保留点阵的两个副本。该程序在一个副本上检查并确定邻域目前的状态，而把计算所得到的结果记录在另外一个副本上。边界条件必须也是确定的。在理想的情况下点阵是无限的，但这显然是不现实的。有一种通用的技术是有效地把点阵的边接在一起，使得对边上的细胞成为相邻的细胞。在一维的情况下，这种点阵拓扑等价于一个圆，在二维的情况下等价于一个环面；虽然它是有限的，但是并没有边界。

上述这种程序是在通用数字计算机上运行的序列过程，它模拟了许多计算机同时运行的行为。如果采用一种许多计算机组成的具

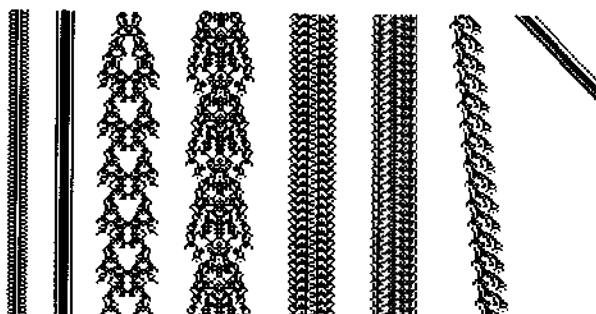


图 5　一种可能的万能计算机的某些组件

有细胞点阵结构的网络,那就要好得多。构造这样的机器决不是不可能的:网络中的每台小计算机都非常简单,因此许多计算机都可以装在一块半导体硅片上。只有相邻的计算机才需要彼此进行通信这一事实也能减低这种装置的复杂性。Toffoli 估计,这种处理机可以比通用计算机快百万倍或者甚至快 10 亿倍。在麻省理工学院和马萨诸塞州 Waltham 的思维机器有限公司已经着手进行有关这类计算机的初步工作。

Toffoli 用标准的微电子元件取代专用硅片制造了一个专用的细胞自动机装置。计算是依次进行而不是对所有的细胞同时进行,但是由于这种装置精巧地调成进行同一类计算,因而它比通用计算机要快 1 000 倍。这种机器本身是由一些装在机架上的印刷电路板组成的,它和彩色显像管联结,并由另一台小型计算机 Atari800 来控制。

Toffoli 的细胞自动机装置具有 256×256 的细胞点阵,每个细胞可有多达 256 种状态。每秒钟每个细胞的状态都要重新计算 60 次。观察以这种速度运行的系统与观察一个低速装置有着根本的不同。这时人们所看到的不是一连串静止的照片,而是一幅运动的画面。于是生命游戏不再表现为抽象模式的静态过程,而更像是从显微镜



里观察细胞和原生物在游动、旋转、繁殖、吞食其他生物或被其他生物吞食。

一维细胞自动机对于计算机资源的要求，从空间和时间来讲都比二维系统要少得多。写出一维系统的程序也要容易得多。直线点阵还有一点比平面点阵优越：由于它的几何结构比较简单，因此更有希望对于这种自动机的发展过程得到一种分析的理解。高级研究所的 Stephen Wolfram 在过去两年所从事的工作就是为了做到这一点。

一维点阵的每一代都是一行细胞，但是相继的若干代可以一个挨一个地画在一起。这样就可以形成一个二维的模式，一根轴表示空间，另一根轴表示时间。于是整个系统的发展过程也就一目了然了。

Wolfram 发现，迄今为止他所研究的所有转移规则可以分为 4 类。第一类规则的发展导致稳定和均匀的状态，例如所有的细胞都取值 0 或者 1。第二类规则产生一些简单的结构，或者是稳定的，或者是周期性的，不过在每一种情况下它们都彼此隔离。第三类规则产生一种无秩序的模式，虽然还不是随机的。第四类包括少数几种转移规则，它们产生的结构在空间和时间方面都是极其复杂的。

Wolfram 猜想一维的细胞自动机也许是被明确定义的具有复杂的自组织行为的系统中最简单的一种。在自然界里，许多连续的动态系统具有这种能力：从随机的初始状态出发可以发展出一种高度有序的结构(雪花便是其中一例)。这种发展过程可以用吸引子来解释，它们似乎把系统引向所有可能结构中的一个子集。

在各种细胞自动机与在物理系统中所观察到的各种吸引子之间已经确立了相似的关系。第一类自动机类似于具有最简单吸引子的连续系统；这种吸引子是一个有限的点，它总是把系统带到同样的最终状态。第二类自动机的发展很像是一个具有有限循环的系统，即

一组结构在无限地重复。

第三类自动机具有无秩序的模式，它可以和更加有趣的所谓奇异吸引子联系在一起。奇异吸引子是诸如湍流突然发生之类的物理现象的特点。在一个受奇异吸引子控制的系统中，发展过程会趋向所有可能结构的一个子集，但是这个子集可能具有非常复杂的结构。把这个集合看成空间中的点阵时，在许多情况下它是分数因次曲线，这是一种具有分数因次的几何图形。

我们考虑一个简单的实验就可以更明显地区分各种自动机。假设一个细胞自动机从某个随机选择的初始构形开始发展，并且让它发展许多步；记下最终的状态。然后再回到最初的构形，只改变一个细胞的值，让该系统发展同样的步数。这种微小的变化将会对最后状态产生什么样的影响呢？在第一类自动机中根本不会有什么影响：不管最初状态是什么，第一类系统都会达到同样的最终状态。第二类系统会产生某些效应，但是限于发生变化的这一点附近很小的区域内。然而在第三类系统中，仅仅变化一个细胞就会引起混乱，并且波及整个点阵。

第四类规则是最罕见的，也是最复

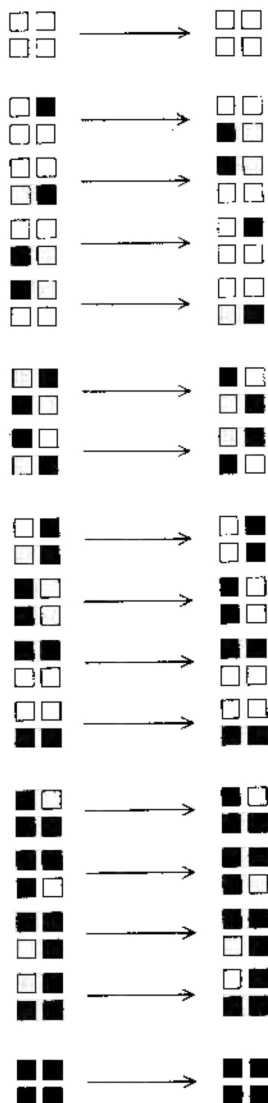


图6 弹子球计算机转移规则



杂的。这一类中也包括某些非常简单的转移函数。例如在一种包括中心细胞和每边两个细胞的邻域中，如果规则规定当有两个细胞或四个细胞为 1 时，中心细胞为 1，则就会导致第四类的模式。第四类系统对于初始条件变化的敏感程度甚至超过了第三类系统。据猜想，要想预测第四类自动机的未来状态，最有效的方法就是让自动机自己去计算这种状态。

一个有关的猜想甚至有更大的范围：它主张第四类自动机也许有资格成为万能计算机。图林机就是这类计算机中最为人们熟悉的一种。一种函数只要是可以计算的，那么据推测图林机就能做到这一点。其他的计算机只要能够表明它与图林机等价，那么就可以证明它也是万能的。已经证明有些二维细胞自动机(其中包括生命游戏)是万能计算机。对于每个细胞有 18 种状态的复杂的一维系统也给出了证明。第四类自动机是已经知道的最简单万能机，它的大部分主要元件都已经确定。还差一个重要的元件，即时钟；这个结构产生一系列固定间隔的脉冲，就像是生命游戏中的“滑翔导弹枪”。

把细胞自动机看作是计算机的看法表明，细胞自动机的自组织行为可以用其计算能力来加以描述。例如，可以把一个细胞自动机的发展过程所产生的构形集合看作一种形式语言。每种构形可以当作这语言中的一个单词，它根据一组语法规则，由代表细胞自动机点上的值的一系列符号组成。Wolfram 证明，任何细胞自动机在经过一段有限的时间之后所产生的构形都可用一类简单的形式语言(称为规则语言)来描述。对规则语言中的任何一种，都可以找到一个最简单的语法。这语法给出细胞自动机构形的最低限度的描述，因而其规模可以看作是对构形的复杂程度的一种度量。对于第一类和第二类细胞自动机，经过充分多步之后，其构形复杂性趋于某一有限极限，因而这些系统所产生的结构可用规则语言加以描述。但对第三类和第四类细胞自动机，其复杂性通常与时俱增，因而看来要描述这

类系统在长时间之后的行为,就需要更为复杂的形式语言。

有一类特殊的细胞自动机是可逆的。对于一台可逆自动机来讲,可以从任何构形出发发展任意多步,然后停止并逆转,它总能恰好回到原来的初始状态。由典型的可逆自动机形成的模式与不可逆自动机形成的模式有着质的差异。特别是,假如开始的模式是随机的,那么模式总是保持随机性;不会出现什么自组织的结构。

可逆性的必要条件是转移规则必须是确定的,不管正向还是反向都是如此。也就是说,一个邻域的每一种可能状态的先行状态或后继状态都必须是唯一的。生命游戏是不可逆的,因为一种状态的先行状态无法唯一地确定。例如,假定有一个细胞现在是"死"的,那么它的前一代可以有任意数目的活邻居,只要不等于3就可以了。Fredkin发明了创造可逆转移规则的系统方法,Margolus则对这方法作了进一步的研究。这种方法的本质是让细胞的下一个状态依赖于邻域的前2个状态。在时间$t+1$的状态由邻域在时间t的某个函数减去时间$t-1$的同一函数来确定。这样逆过程就是非常简单的,在时间$t-1$的状态一定是由时间t的状态减去在时间$t+1$的状态来确定。

由于双向确定性的要求,在可逆自动机的发展过程中就不可能有吸引子。吸引子的存在意味着有许多初始状态在发展的途径中会彼此合并,因而在相反的过程中这些合并的点就成了分叉的点,在这些地方就不再是确定的了。同样,可逆的细胞自动机永远不能进入或者离开循环,因为这样一来也会在这个方向或那个方向上出现分叉点。由于排除了吸引子和有关的自组织模式,可逆的转移规则似乎是产生了一种枯燥无味的细胞自动机,但是这类系统的其他性质可以提供有趣的补偿。最值得注意的是,在可逆自动机中细胞模式的信息量可以证明是一个守恒量(它在自动机的发展过程中既不会增加也不会减少)。这种性质使得可逆系统成为有价值的计算模型。



Margolus 根据 Fredkin 最早讨论的一种假想机械系统(弹子球计算模型)构造了一个细胞自动机计算机。在这种模型里,信息单位(1 和 0)是由理想的弹子球来携带的,这些球无摩擦地运动,而且彼此之间或从障碍物上的反弹都是完全弹性的。在指定位置有一个球就表示二进制的 1,没有球就表示二进制的 0。巧妙地安排撞击物就可以产生各种逻辑门,它们类似于电子计算机中相应的元件。例如在一个与门,只有当两个球沿着专门轨道同时到达这与门时,才有一个球通过输出区域(从而记下一个二进制的 1)。

Margolus 的基于弹子球模型的细胞自动机是一种简单而有点不寻常的可逆转移规则的例子。细胞并不是单独加以考虑,而是 4 个组成一块来考虑;在块内的每一种模式都变换成为唯一的乘积模式。转移规则使得单个的 1 在 0 的背景中沿着点阵的 4 个对角线方向中的一个传播,其速度是每一步移动一个细胞的距离。这个孤立的 1 就相当于弹子球。由 4 个 1 组成的实心块永远不变,它起完全反射器的作用。把这种模型放在 Toffoli 的细胞自动机计算机上运行时,这些"弹子球"就会飞速穿过显示屏幕,表现出复杂交织的模式。看到这种有序的(虽然也是狂乱的)运动,很难设想这种程序并不是说明球的径迹,而只是把一条规则应用于所有的细胞。

弹子球模型及其在细胞自动机上的实现对于计算理论具有重要的影响。人们猜想任何计算机都必定有同时耗散能量和信息的元件。按照这种观点,计算机的效率和热机的效率一样具有热力学的极限。人们所认为的信息和能量的这种不可避免的损失,其直接原因是由于计算过程的不可逆性(3 加 5 等于 8,这个过程是不可逆的,因为有无限多的不同数目相加可以得出同样的结果)。

Fredkin、Toffoli 和 Margolus 指出,弹子球模型提供了与必然会有耗散这一观念相反的论据。在弹子球计算机中没有信息的损失。实际上,弹子球本身是无法创造或者消灭的,因而在系统的发展过程



中规定它们初始模式的所有信息是守恒的。只要追溯弹子球的轨道就可以使加法运算的输入恢复。从原则上讲，弹子球计算机可以不消耗内部能量而进行操作。

Toffoli 的一段话特别清楚地表明了物理学和计算之间的联系。我们可以把这段话理解为描述了最大的细胞自动机。他写道："几十亿年来，自然界都在不停地计算宇宙的'下一个状态'；我们必须做的，实际上也是我们唯一能够做的，就是跟着这正在进行的硕大无比的计算过程前进，并企图发现它的哪一部分恰好趋近我们所希望的地方。"

34. 奇异的混沌吸引

了解混沌内在特性的人都有这样的概念，即混沌之中存在一种奇异的吸引力。某些表现出混沌行为的物理系统中也有这种吸引力，其原因是那些系统在某种程度上被吸引到了混沌的模式中。混沌模式本身就像奖金一样，也有着奇妙的吸引力。有些读者可能已经知道，构成混沌行为的基础的是一些叫作奇异(或混沌)吸引子的几何形式。奇异吸引子能够在家用计算机上生成。

读者们必须具备物理直觉，才能理解下面要讲的东西，特别是理解什么是吸引子。那么，什么是吸引子呢？粗略地讲，吸引子产生于平衡的概念。它是一个系统的行为的归宿，或者说是系统的行为被吸引到的地方。用单摆这种简单的物理系统就可以对吸引子这个概念加以具体的解释。假设有一个摆动着的普通单摆，由于摩擦力的存在，它的摆动最终会停下来。人们借助于所谓的相图(或状态图)就能够描述单摆的运动。在相图上绘出了摆与垂直方向构成的角度与该角度的变化率之间的关系。摆的动态特征可以由相图上的一个围绕原点按环形路径运动的点表示出来。随着摆的能量的减少，该



点的运动轨迹呈现为一个越来越靠近原点的螺旋形，在原点处摆就完全停止摆动。在这个例子中，原点就叫做吸引子，因为它似乎在吸引相图上那个描述单摆运动特征的点。读者们如果认为这种只有一个点的吸引子并没有什么奇异的地方，那完全是正确的。

落地大座钟的摆的运动的吸引子要稍微复杂一点。这种大座钟有一个摆轮装置把机械能输送给钟摆，使它的摆动不会慢下来。如果把钟摆猛力一推使钟开始走起来，那么钟摆的摆动速度会渐渐变慢，最后稳定在摆轮规定的速度上，此后速度不会再降低。如果开始推钟摆时用力太轻，钟摆就会像普通的单摆那样逐渐地停下来。在前一种情形，相图上描绘出来的是一条越来越靠近一个圆形轨迹的螺旋线。因此，这个系统的吸引子是一个闭合圆环。从这点来讲，一个圆环也并不比一个点更奇异。

如果给普通的摆钟引入一个垂直方向上的振动，就能使摆的运动表现出混沌行为。为了产生这种振动，我们用一个电动机使摆的支承点作正弦型的上下运动，这样摆就会开始疯狂地摆动，不存在任何周期性行为。

但是，在此我要选择另一个物理系统来介绍混沌概念。设有三个放大器。第一个放大器输出信号 x 给另两个放大器。第二个放大器收到信号 x 后将输出信号 $1-x$。第三个放大器接收信号 x 与 $1-x$。产生出这两个信号的乘积 $x(1-x)$，并把它反馈给第一个放大器。第一个放大器在接收 $x(1-x)$这一输入的同时还接收一个控制电压 r。另外还有一个附加元件（它能在很短的时间内对它的输入信号进行采样并把这一输入电压作为输出传送出去）使电路闭合，该附加元件插在第一个放大器的输出引线上。这种由三个放大器组成的电路产生出一个电压振荡，随着控制电压 r 逐渐增大，电压振荡愈加剧烈。

当 r 小于 3，且选择的 x 的初始值不为 0 时，电路在经过短时间



的振荡后就稳定在一个特定的 x 值上而不再变化。这个 x 值就构成一个单点吸引子。如果把控制电压 r 增高到 3 以上，电路就在两个 x 值之间振荡，此时的电路叫做双稳态电路，它的吸引子由两个点组成。当 r 的值进一步增大时，电路可以在四个点上振荡，进而可以在八个点上振荡。随着控制电压 r 的不断加高，吸引子的点数就不断翻番。当电压大约达到 3 与 4 之间的中点时，电路突然出现紊乱，它以电子速度无休止地搜寻标志它的先前的存在的那种简单循环形式。此时电路行为处于一个无形的具有无穷多个点的奇异吸引子的控制之下，这样就出现了混沌。

我选择的这个电路可能会引起读者的兴趣。具备电子学知识的读者可以自己制作这样一个电路，而其他读者可以在任何大小的计算机上模拟这种电路，从显示屏上清楚地观察上述振荡。只需编写出一个简单的程序来计算迭代方程 $x \leftarrow rx(1-x)$ 就能模拟这个电路。下面就是这样一个程序（我称它为 CHAOS1），它的核心部分是六条指令：

```
x←3
对于 i←1 至 200
x←rx(1－x)
对于 i←1 至 300
x←rx(1－x)
绘出(200x,100)
```

变量 x 的初始值规定为 3，然后 CHAOS1 程序进入第一个循环。在该循环内，上述那个基本的迭代公式要迭代 200 次，从而使中间值逐渐消失。中间值是迭代方程自身在迭代过程中必然有的，不是不精确的运算造成的。其原因我在后面将用几何方式解释清楚。接下来程序进入第二个循环，在这个循环内，迭代方程要迭代 300 次，每次都要绘出 x 的值。

绘图指令中的数字100没有什么特殊之处，此时屏幕的尺寸假定为200×200。横坐标是200x，它使计算出的各个x值（始终在0～1之间）散布在屏幕的一行上，该行的高度为100，即位于假想的屏幕的高度的一半上。

核心程序或者是把一个点绘出300次，或者是绘出几个点，每一点的绘出次数少于300次，这取决于你规定的控制电压r的值是多大。它甚至可能通过绘出奇异吸引子的300个不同的点来设法"截获"混沌。如果迭代次数增加，该奇异吸引子看上去就更清楚。不管迭代多少次，只要迭代过程一进入稳定状态，x的值就按一定的规律从该吸引子的一个点跳至另一个点。奇异吸引子的点数不论是有限或是无限，它们都可以叫做"轨道"。

如果程序计算出了大量的这类图形（每一图形都在前一图形的下面），一张完整的简单放大器电路的状态图就呈现出来（见图1）。把2.9～4之间的值从小到大代入迭代方程中的r，就能产生出那些图形。比如以屏幕顶端为2.9，底部为4，中间分成200个代表r的值。r取值越多（比如4 000个），图形就越精细。但取值太多不适合于屏幕显示，只能作为一个整体绘出。

当r的值小于3.56时（更精确地说应是3.569 945 718 69），迭代方程$x \leftarrow rx(1-x)$所体现的简单动态体系的吸引子只有几个点。这些代表非混沌状态的点排列在三条大区带和无数条小区带中。当r值渐渐逼近3.56时，吸引子的行为开始变得奇异起来。当r值达到3.56时，一直是光滑的分歧线突然散开变成一些混乱的点，于是混沌就出现了。还有一个奇怪的现象：在r值不断接近4的期间，混沌特征会不时地消失掉。

状态图完全作好后称为分歧图。从侧面看上去，这张图就像一颗什么星的光谱，分布着一些曲线和一些很漂亮的阴暗折皱。为什么分歧图上会有这些装饰性的东西，其原因是很复杂的，只有混沌理





图 1　分歧图上显示出混沌出现的过程

论才能解释它。后面我对这个问题还要进行更深的探讨。眼下还有一个大多数读者更为关心的问题:为什么看上去很普通的一个方程却能产生出如此不可思议的结果?

绘出方程 $y=rx(1-x)$所规定的抛物线,就能够从几何上对那个迭代方程在 r 取非混沌值时的行为加以模拟。如图 2 所示,该抛物线方程中的 x 为横坐标,y 为纵坐标。画一条对角线 $y=x$ 迭合在抛物线上。r 的取值为 3.3,当 r 取这个值时该系统的吸引子为两个点。为了表现出该系统的动态特征,先要确定 x 的初始值。我选择 0.3 为初始值。当然,也可以确定为别的数字。

从图的底部 x 为 3 的那一点起,向上画一条垂直线直至与抛物

线相交，这样就模拟出 $x \leftarrow rx(1-x)$ 的第一次迭代。我把该垂直线与抛物线相交的那一点记为 A 点。该点的高度即为相应的 y 值。在作第二次迭代时，这个 y 值又作为 x 的值代入方程。从图上看，这个模拟过程就是测量垂直线的高度，把它标在横坐标轴上，然后从这点再画另一条垂直线与抛物线相交。有一种比较简捷的办法是从 A 点向斜线 $y=x$ 画一条水平线；由此产生出又一个交点就标为 B 点。注意，B 点和原点位于边长为 y（由第一次迭代得出）的一个正方形的对角线两端点上。因此，从 B 点至抛物线画一垂直线（交点标为 C），就能把 y 值反馈给这一系统。这样，反复地向抛物线作垂直线直至其与抛物线相交，然后向斜线作平行线直至其与斜线相交，就得到一条逐渐收缩于一个正方形的矩形路径。

这个几何方法类似于 CHAOS1 程序的核心部分。最后所得的正方形与抛物线相交的两个点相当于两点型吸引子的两个点。富于探索精神的编程者也许能够编出程序使计算机产生出这样的图形来，这样他们就能够对所研究的那个“简单”的迭代方程获得透彻的理解。尤其是，当混沌开始时，那幅图会是什么样子呢？产生混沌的的 r 的各个值所给出的看来是随机的数字真的是随机数吗？

方程 $x \leftarrow rx(1-x)$ 的迭代相当于把 0 至 1 之间的点映射到一条抛物线上。在这个单位区间上紧密地挨在一起的那些点被映射到抛物线上后，相互隔得很开，接近于 0 的那些点更是如此。当然，这种情形要用 $rx(1-x)$ 代替了 x 时才能出现。折迭操作得以进行，是由于抛物线具有左右对称性。除了在抛物线的顶点之外，单位区间上始终有两个点映射到同一个值 $rx(1-x)$ 上。这两个点当然就是 x 与 $1-x$。

人们已经用混沌理论对分歧图的大部分结构进行了分析。混沌区域的边界是由当 $x=0.5$ 时的迭代值中，迭代次数的最小值和最大值确定的。这些最小值和最大值所构成的曲线，以及那些奇怪地罩



在混沌区域上的“纬幕”所构成的曲线全部都是含 r 的简单多项式。阴影最深的地方就是穿过这些阴影的奇异吸引了的点的密度最大的地方。那些空白区带表明该区域内有序代替了混沌。理论告诉我们，对于每一个整数，都有这样一条区带(无论是多么的窄)，该区带由其数量恰好等于这个整数的若干条轨道构成。最后，奇异吸引子(即使是上面讨论的那个简单系统的奇异吸引子)都有一种分数维性质；无数的点能在所有放大水平上显现出引人入胜的细节来，Mandelbrot 集一样。对于这一点，熟悉混沌现象的读者是不会感到诧异的。

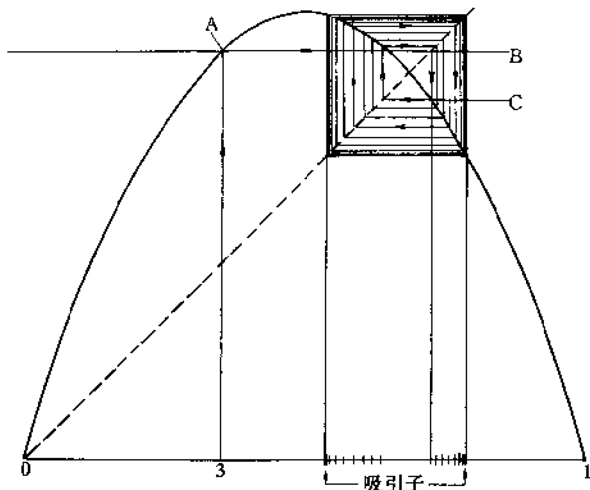


图 2　对一个简单系统作几何模拟产生出一个两点吸引子

以法国数学家 Michel Hénon 的名字命名的一些方程可以描述更为复杂的动态系统的行为。所谓的 Hénon 映射不仅能描述像移动小行星和滴水龙头这样的物理系统，同时还能在映射过程中产生出美丽的图案。Hénon 映射不是一个方程，而是两个方程。下面即

是一例：

$x \leftarrow x\cos(a)-(y-x^2)\sin(a)$

$y \leftarrow x\sin(a)+(y-x^2)\cos(a)$

两个方程右边的变量 x 和 y 的当前值用来产生方程左边的新值(也用 x 和 y 表示)。

一个叫做 CHAOS2 的程序利用了这两个方程来产生出许多动态系统内固有的有序和混沌的图案。CHAOS2 的核心程序与 CHAOS1 的相似：

输入 x 和 y

对于 i←1 至 1 000

$xx \leftarrow x\cos(a)-(y-x^2)\sin(a)$

$y \leftarrow x\sin(a)+(y-x^2)\cos(a)$

x←xx

绘出(100x,100y)

这两个程序有两点不同之处：CHAOS2 的迭代变量是两个而不是一个，且 Hénon 映射描述的系统是保守的而不是耗散的。由于用了两个变量，因此当第二个方程仍然要用到 x 的当前值时，你就不得不用中间变量 xx 来表示新的 x 值。由于这个动态系统是保守的，用来消除瞬态值的那个初始迭代循环就可以去掉不要。这种保守系统不存在由摩擦或别的耗损性泄漏而引起的能量损失。因此，这样的系统也就没有这种吸引子。然而可以这样说：该系统计算出的每一条轨道就是这系统自己的吸引子。无论如何，Hénon 映射中肯定存在奇异性(及混沌)。最后，对参数 a 的每一给定值，所得的系统都有大量的轨道；而且，由于这种系统的保守性，每对 x 和 y 的初始值都代表了已经出现在一条轨道上的一个点，换句话说，即吸引子是瞬时的。由于这些原因，CHAOS2 的核心程序没有使用标准初始值来作为迭代变量。迭代变量必须由编程者自己输入计算机。



编程者把一个确定变量 a 的值的输入语句放在 CHAOS2 程序的核心部分之前,该程序就算最后完成了。CHAOS2 与 CHAOS1 一样,每个新的 a 值都产生一个新系统。但由于这些系统是二维的,对轨道曲线的抽样势必要用掉所有能用的空间;所以你无法系统地改变控制参数 a,而不引起不期望的混沌出现。

因此,CHAOS2 的使用者先要规定一条初始轨道,其办法是把该轨道上一个点的坐标打入计算机,然后他就可以坐下来津津有味地观察计算机把轨道绘制出来。绘制出来的可以是一条曲线(间断的而不是连续的曲线),也可以是别的什么更奇怪的图案。例如,图3是 a 值为 1.111 时的 Hénon 映射图,此图上有 38 条轨道。从图的中心向外,开始是一些一条套一条的闭曲线,然后突然出现一些"小岛",即若干独立的轨道夹在这些闭曲线之间。一条套一条的轨道一直向外延续到混沌出现的区域。在这张相图外围出现的"岛屿"更多,同时还出现了一些闪烁不定的点子,这些点子的出现表明有混沌存在了。图中用矩形框出的混沌区域放大后在旁边示出。我要告诉那些打算把 Hénon 映射图放大的读者,在放大这种图时应当用他们的计算机上所能用的最精确的算法。

正如我前面提到的那样,Hénon 映射可以描述各种保守系统,比如绕太阳运行的小行星这样的系统。遗憾的是,映射图上的轨道不是小行星真正的运行轨道,而是那些轨道的相图。在刚才描述过的图中,横轴可以表示出行星在其运行轨道上的位置,即到太阳的距离。纵轴可以表示出行星在该位置上的径向速度。由 Hénon 映射计算出的轨道上的每个点表示行星在处于相对于太阳的某个特定角位置上(也就是当行星穿过与太阳成该角度的一个垂直平面时)的径向距离和速度。由映射计算出的相继各点,代表行星在该平面上的相继出现。上面提及的"岛屿"是一些共振带,是由太阳系中的大星体(如木星)对小行星轨道的摄动引起的。在混沌区域内,当小行星



每次回到原来特定的平面上时，它的径向位置和速度都会发生改变，且这种改变基本上是随机的。它的运动无法预言，什么情况都可能发生。

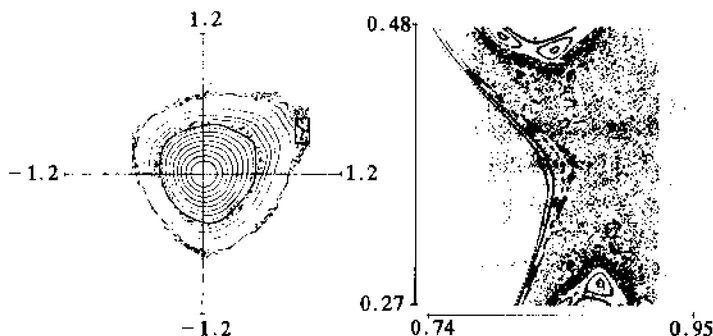


图 3　Hénon 映射中的相继的轨道(左)退化成混沌(右)

Peter de Jong 提出了另外一些能产生出希奇古怪的形状和图案的迭代公式。他提出了有四个参数的迭代方程，即 $x \leftarrow \sin(ay) - \cos(bx)$ 和 $y \leftarrow \sin(cx) - \cos(dy)$。x 与 y 的初始值都为 0。读者可把 $a = 2.01, b = -2.53, c = 1.61, d = -0.33$ 代入方程运算。看是否能得出 de Jong 称为“鸡腿”的图案。把 $a = -2.7, b = -0.09, c = -0.86$ 及 $d = -2.2$ 代入方程可产生出“圆点发射器”。把 $a = -2.24, b = 0.43, c = -0.65$ 及 $d = -2.43$ 代入方程可产生出“自装饰的圣诞彩蛋”。

35. 三维生命游戏

生命游戏(Life)本来是一种在二维方形网络上进行的细胞自动机游戏,但现在它已启发人们发明了与其类似的三维空间里的生命游戏。Carter Bays 探索了许多种三维空间的生命游戏,并且发现其中有两种最有前途,他把这两种形式命名为 Life 4555 和 Life 5766。它们都再现了最初的生命游戏的诸多特征(如像闪光灯及滑翔物等);而其中之一无疑将能够与 John Horton Conway 在 1968 年发明的生命游戏成为一对当之无愧的伙伴。

该游戏是在一个无限的二维正方形方格网上展开的。每个单元(即方格)有八个邻居(四个在其周围的四角上,四个与其四条边相邻)和两种存在状态,即"活"或"死"。一只大钟在某处嘀嘀嗒嗒地走着。每嘀嗒一声,都可能有一些单元变活,而另一些单元死去。每个单元的命运由其活着的邻居数决定。例如,在某一时刻,如果一个活单元有少于两个或多于三个的活邻居,那么这个单元在时钟响下一嘀嗒时就会死去,其原因是这个活单元处于营养不足或过分拥挤的状态。另外,一个死单元如果正好只有三个活邻居,它在时钟响下一嘀哒时就会起死复生:诞生一个活单元需要有三个"母单元"。

Conway 把他的游戏称作 Life 是因为那些单元可以处于"死"或"活"两种状态。人们很快发现,这个名称比预想的还要恰如其分。由活单元组成的各种各样的结构呈现出惊人地复杂而且栩栩如生的形态(见彩图 10)。这些形态是循环出现的:当时钟每响一下时,这些形态就发生变化,但在响了一定数目的嘀嗒声之后原先的形态又会重新出现。有些形态保持在它的位置上不移动,而另外一些则沿水平方向、垂直方向或对角方向每次移动一个单元,逐步穿过网格。这两种类型的形态各有一些古怪的名字。静止形态的名称有灯塔、蜂房、闪光灯及方块等(见图 1)。移动形态的名字有滑翔物及飞艇等。然而 Conway 的生命游戏远远超过了一般意义上对自然现象的模拟;在生命游戏的单元平面上甚至有可能建造出一台计算机。

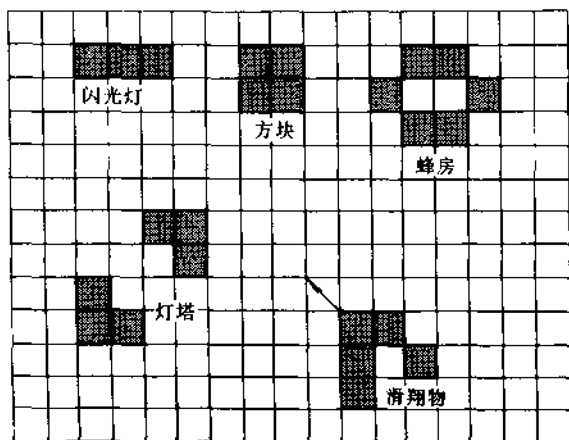


图 1　Conway 的二维生命游戏中的几种形态

在宽敞的三维单元空间内,Conway 的生命游戏将会产生更为古怪的现象,这是不奇怪的。Carter Bay 的 Life 4555 和 Life5766 就是这种三维空间内的游戏。它们的每个单元是一个立方体而不是正方形,它有 26 个邻居而不是 8 个邻居。



Life 4555 和 Life 5766 这两个名字出自于 Bays 所提出的一类专用术语。头两个数字表示活单元继续存在的条件，第一个数代表一个活单元要避免营养不足所必须保持的活邻居的最小数目，第二个数则代表要避免过分拥挤一个单元能够容忍的活邻居的最大数目。后两个数决定了死单元的命运。第三个数表示一个死单元为了起死回生所要求的最少活邻居数，第四个数则是这个死单元为了复活所能容许的最大活邻居数。按照 Bays 的表示法，Conway 的生命游戏就成了 Life 2333。

Life 4555 的运行和 Life 2333 完全一样简单，如果一个活单元有少于 4 个或多于 5 个的活邻居，它就会死掉。如果一个死单元正好有 5 个活邻居，它就会起死复生。在对这样的一系列规则作一般的考察时，Bays 注意到一个由立方体单元组成的奇特构型慢慢地从他的 Macintosh 计算机显示屏的深处蠕动出来(见图 2)，于是 Life 4555 第一次引起了 Bays 的注意。这个立方体单元组成的结构是一个三维的滑翔物，它依次显示出四种不同的图案，然后又重新开始。

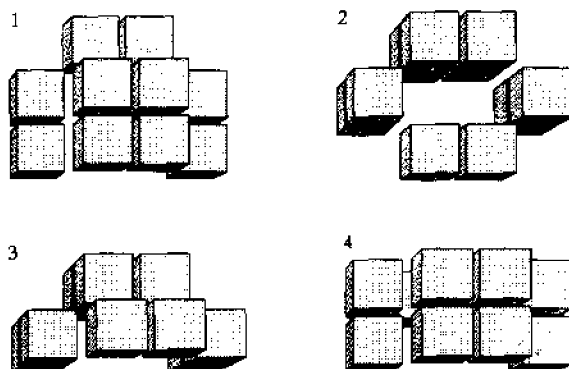


图 2 Life 4555 中的滑翔物

每种图案由 10 个排列成椭圆形的立方体组成，但这些椭圆被弄出一

些奇形怪状的钝口,像自由降落的长沙发一样穿过屏幕。

好奇心促使 Bays 决心通过一系列"原汤"实验来进一步探索 Life 4555 的规则。他在起始空间中随机地选定一些立方体单元并使之成为活单元,然后使这个单元宇宙开始运转起来。每一代都有一些立方体死掉,而另一些立方体变活。活立方体数目一代一代地减少,然而 Bays 也注意到了一些奇怪的稳定立方体集合,它们并不随代的推移而发生改变。其中一些使他联想起支撑架、十字架、台阶、球体和杠铃的形象。随后的实验产生了其他一些稳定的结构以及许多循环稳定结构。Bays 给它们取了像"转子"及"狂跳的野马"这样一些古怪名字。

Bays 像核物理学家研究新的基本粒子那样,让滑翔物及其他的小型结构在一切可能的角度上发生相互碰撞。其中最令人惊讶的碰撞之一是某种滑翔物与球体间的碰撞,这一碰撞导致了一个混乱群体的产生,它随后膨胀成了一个有 29 个立方体的结构,这一点本身并不特别值得注意。但是,那个滑翔物在几代后又突然出现了,只是其位置略微移动了一点。

"原汤"实验的产物以及滑翔物碰撞的结果被 Bays 称为"自然界"。有些形态是很容易得到的,而另一些更奇特的结构就要花费一番功夫才能得到。例如,一些拱状的结构可以连接在一起形成一个新的稳定结构,Bays 称之为拱廊。一个由栅栏、楼梯、墙和链子组成的完整建筑物呈现出来。墙能够被弯曲成螺旋状和其他许许多多希奇古怪的稳定形状,其花样之多仅受热衷于 Life 4555 的人的想象力的限制(见彩图 10)。

还有其他一些现象不能在此一一描述。有一种名叫"greeter"的由七个立方体构成的形态生性孤僻,只要有一另个 greeter 在场它就会死去。一个滑翔物在滑翔途中经过 greeter 时一定会被它抓住不放。第二个滑翔物要与该 greeter 碰撞并使它爆炸才能侥幸救出它



的同胞。

在 Bays 发现的两种卓有成效的三维生命游戏中，看来他更宠爱 Life 4555。非常奇怪，只要给 Conway 的生命游戏 Life 2333 中的每个数字加一个 2 就能得出 4555 这个编码。这种巧合也许预示了 Life4555 最终会与 Conway 的游戏成为一对伙伴。

但是，严格说来，Life 5766 比 Life4555 更酷似 Conway 的游戏。特别是在某些特定条件下，Life 5766 就更像平面上的 Conway 的生命游戏。这些条件是 Bays 发现的一个定理所规定的。

可以这样想象一下：目光向下注视着 Conway 的生命游戏的网格平面，并假定平面上有一个由活的（正方形）单元组成的结构。在每一个活的正方形单元的顶面及下方各放一个活立方体。根据 Life 5766 的规则，当且仅当下面两个条件被满足时，这些立方体将始终完美地模拟夹在中间的 Conway 游戏的那些单元的行为：

1. 网格平面上的任何活单元永远没有 5 个活邻居。

2. 网格平面上的任何死单元永远没有 6 个活邻居。

生命游戏中的许多形态（包括 Conway 的滑翔物）都满足这两个条件，把一个二维滑翔物变为立方体结构的形式，就可以创造出一个 Life 5766 的滑翔物。在三维空间内这个滑翔物占据了两个相连的立方体单元层，并被永远限制在此范围内运动。

Conway 的生命游戏中的其他许多结构，如像灯塔、蜂房、闪光灯、方块以及更古怪的小艇、时钟及理发店的宣传招牌等，都满足 Bays 定理所规定的这两个条件。它们都作为 Life 5766 的实体，以两层构造的形式存在，而其行为则与两维空间的对应形态完全相同。Conway 的生命游戏里的各种形态并非都满足这些条件，其中的害群之马就是著名的滑翔物枪。如果在滑翔物枪生命的任何阶段上对它作一个检查，就会发现一些有 5 个活邻居的活单元。违反了上述两个条件中的任何一个，都会使 Life 5766 不再模拟 Conway 的生命游



戏。Bays 指出："当这种情况发生时，一直被限制在两个平面上的滑翔物枪几乎总是形成一个略显球形的三维体，通常很快就死去了。"

应用 Bays 所谓的时空屏障能够使 Life 5766 更为完美地模拟 Conway 的生命游戏。该时空屏障由一片有孔的活单元板组成。孔的分布方式使得屏障上的每个立方体正好能有 7 个活邻居，这样它们刚好不会因过度拥挤而被挤死，然而紧邻时空屏障的任一立方体都将有六个以上的活邻居，所以永远不会变成活的。于是时空屏障的两侧就形成了死区，可以用来精确地再现 Conway 的生命游戏。可以建立两层平行的时空屏障，它们之间相距四个立方体，这样两层屏障之间夹着四层平面。中间的两层平面可以产生任何形式的生命游戏形态，因为此时占据这两层平面的任何一个活单元结构都服从 Bays 定理的两个条件。

尽管 Life 5766 能以某种形式模拟 Conway 的生命游戏，但它终究缺乏一种可以称为"单元活力"(Cellular Pizzazz)的东西。据 Bays 说，Life 5766 的"原汤"似乎总是比 Life 4555 更快地"平静"下来。Life 4555 的好动特性表明了它在计算过程中出现的各种可能性更多。它的确也有许许多多的稳定结构和对称的振荡结构。Life 5766 与 Life 4555 谁能成为 Conway 的生命游戏的伙伴也许最后要取决于能否使 Life 4555 模拟二维生命游戏。如果能做到这一点，它就在各方面都比 Life 5766 优越。

编写可以计算和显示这两种三维生命游戏的连续各代的程序是很简单的，至少在原理上是如此。这种程序甚至可以经修改后用来搜寻规则，以期发现可能被 Bays 漏过了的第三种三维生命游戏。

赋予两个称为 cells 和 newcells 的三维数组以三个下标 i、j、k。它们与单元空间上的三个坐标对应。数组中每一项的内容表示该项所对应的单元是活的或是死的。规定 1 表示活单元，0 表示死单元。

为了计算每个单元在每一代中的状态，无疑需要有三个嵌套循



环。最外层的循环是用下标 i 计算出扫过单元空间的各平面。在这个外层循环中是另外两个分别以 j 和 k 为循环变量的循环。j 循环计算每个平面上连续的行，而 k 循环计算每行内的连续各单元。读者可以用下面这种普通格式作为指南：

i = 1 至 30

j = 1 至 30

k = 1 至 30

计算邻居数

决定状态

显示活立方体

当然 30 这个数是任意的。但是，只有对那些有极好耐心的读者我才建议他们去尝试比 30 更大的数字，因为这时计算时间会长得多。

程序最里层的那个循环要完成三个基本任务。"计算邻居数"这一任务要求程序检查每个单元的 26 个邻居并数出目前的活邻居总数，它可以通过用三个小循环或排列出 26 个单元全部可能的坐标来完成。采用小循环时，可以用下面的算法：

tot←0

l = i − 1 至 i + 1

m = j − 1 至 j + 1

n = k − 1 至 k + 1

如果 cells(l, m, n) = 1

那么 tot←tot + 1

tot←tot − cells(i, j, k)这过程的最后一行保证了单元(i, j, k)的状态不会加到总数中去。

确定了活邻居的总数后，程序就要确定当前单元，即 cells(i, j, k)的新状态。要完成"确定状态"这一任务只需检查 tot 的大小与 cells(i, j, k)的状态间的关系。

如果 cells(i,j,k)=0

并且 tot=5

那么 newcells(i,j,k)←1

否则 newcells(i,j,k)←0

如果 cells(i,j,k)=1

并且 tot<4 或 tot>5

那么 newcells(i,j,k)←0

否则 newcells(i,j,k)←1

这里,我假设读者们编写的是用于 Life 4555 的程序。读者也可以把上面算法作一些修改,以适用于 Life 5766,或者作出相当一般的算法,以便能应用于任何三维规则。谈到这里,讲点题外话看来是值得的。

前面谈到的状态计算一般要用 4 个变量,Bays 称它们为 el、eu、fl、fu。字母 e 和 f 代表环境和再生能力,而 l 和 u 代表上下限。因此,el 和 eu 是一个单元在其环境中能够继续生存下去的上下限;如果围绕一个单元的活立方体数大于或等于 el 而小于或等于 eu,它就可生存下去。同理,fl 和 fu 是一个死单元的再生条件;如果环绕一个死单元的活立方体数大于或等于 fl 而小于或等于 fu,它就会复活。因此,算法通式为:

如果 cell(i,j,k)=0

并且 tot<fl 或 tot>fu

那么 newcells(i,j,k)←0

否则 newcells(i,j,k)←1

如果 cell(i,j,k)=1

并且 tot<el 或 tot>eu

则 newcells(i,j,k)←0

否则 newcells(i,j,k)←1



至此，两种三维 Life 游戏的程序就可以借助适当的三重循环把 newcells 中的内容移送至 cells 上，而 newcells 就空出来留作计算下一代活立方体之用。

计算过程的最后阶段是"显示活立方体"。如果某一立方体是活的，程序就使它显示出来。绘制实际的立方体时，最好是把它们的可见面都画上颜色。如果只是画出立方体的框架，其结果只会得到杂的彼此挨得很紧而无法辨认的图形。为了使前面的立方体能遮盖住后面的立方体，最简单的方法是使外循环变量 i 从单元空间的后部向前(即向着观察者)扫描。用这种方法必定会遮住一些立方体，这是任何形式的三维生命游戏都固有的唯一的不足之处。我们不能看到三维生命游戏中的发生的一切情况，这一点是跟 conway 的二维生命游戏不同的。同时，任何三维游戏都具有的这一特点也是真实的三维空间的一种性质。我们不能看到正在发生的所有事情，这是非常幸运的。

用程序来实现这个计算过程的最后一步时，其速度显得有点慢。可以使这一步的速度加快一点的简单方法是用球体(实际上是大小随其在屏幕内的"深度"而变的实心圆盘形)代替立方体。此外，Bays 还叙述了许多能使速度提高几个数量级的改进方法。

36. 分数维的山峰与植物



大厅的灯光暗下来，帷幕徐徐打开，银幕上出现了根据J. R. R. Tolkien 的三部曲"Lord of the Rings"所改编的电影。Frodo 在一个开阔的峡谷里溜达着。远处，锯齿状的冰雪覆盖着的山峰耸入云端。近处有些不知是什么种类的奇花异木在阳光下闪烁。转眼，屏幕上的奇景变成了一个男巫凝视着一只水晶球，在这球体的中央出现了一个堡垒，火焰正从它的城垛里窜出来。

虽然现在还很难说 Frodo 是否会在这样的电影里出现，但我肯定那些山峰、树木、水晶球以及火焰都会奇妙地出现在银幕上。这个成就主要将归功于 Pixar 公司（即从前的 Lucasfilm 计算机绘图实验室）所开发的软件和硬件。有家用计算机的读者都能够在计算机上作出基本类似于这些东西的图形来。由于本文篇幅所限，不能在此对水晶球和火焰作一个广泛深入的论述，但还是能够揭示产生它们的基本原理。

在上面描述的假想的电影中，我们可以把摄像机移向 Frodo 身后的那些山峰上。人们可能从来没有见过比这些山峰更令人生畏的



大片陆地了。每一个大的山峰都由一些较小的山峰构成，而这些较小的山峰又由比它们更小的山峰组成，如此下去就形成了一种小山峰的无穷回归。即使一个有皮质脚的滴水嘴一样的海怪站在这样一个犬牙般的地方也会感到难受(见彩图 11)。

原则上，这样的一种山的图形是容易作出来的。为简便起见，我假定这山覆盖了一个三角形的地面。找出每条边的中点，用三条线段把这三个中点连起来。就把这三角形分成了四个较小三角形。用同样的办法再分这四个小三角形。这一过程不断进行，直到达到分辨率极限或计算时间极限为止。结果是得到一大堆令人感到枯燥无味的三角形。如果要使这图形变得生动一些，可以在作图过程中加进一条有关垂直方向上的规则：每当新的中点画在图上时，就使其向上或向下移动某一随机量。通常这个随机数必须随三角形的逐渐变小而减少。这一规则把那些三角形变成弄皱了的山峰和褶皱(见图)。

为什么这一种方法会作出那样逼真的山峰图案呢？答案在于这个过程中产生了一个分数维图形，即当图案不断放大时会显露出更多的细节的图形。分数维形态在自然界似乎是随处可见。我们可以用一个关于海岸线的例子解释分数维图形的基本概念。假设我们要用一根 1 000 米长的测量杆测量出法国海岸线的长度，那么就得沿着海滩向前一杆一杆地进行艰难的测量，同时数出有多少个 1 000 米。然而这样会把许多小的海湾和海岬遗漏掉，所以用这种办法测出的最后得数是不那么准确的。用一根 1 米长的测量杆重复这一过程，会得出一个更精确、数字更大的结果。但即使如此，也有大量的小海湾和岬地被遗漏掉了。无疑，用一根 1 厘米长的测量杆结果就会更为精确。

一般规律是，当测量杆变小时，测出的海岸线长度会增大。测出的长度与测量杆杆长之比率为一个专门值，这个值称为分数维。分



数维与通常说的维不同，它往往被表达成一个分数，而不是一个整数。例如我们讨论的海岸线的维数可能就是一个 3/2 的分数维。可以把这样的一种形状想象成一个介于一维形状(直线)和二维形状(平面)之间的中间形状。如果海岸线比较直，其分数维就接近于 1。如果海岸线很曲折，其分数维就接近于 2，此时它几乎填满一个二维平面。

自然界的分数维模型实际上隐含了细节的无穷回归。从计算机绘图的角度来看，无穷回归是无关紧要的问题；只要景物看来是具有各级放大水平上的细节就行了。在达到屏幕分辨率的极限之前，计算机上生成的山的特征就与上述分割过程中最终所得的三角形的特征一样精细。完整的山峰绘制算法太长太复杂，无法在此作足够详细的介绍。但有一个简单的程度可以绘出 Mandelbrot 峰的断面，它称为 MOUNT AIN。该程序体现了沿垂直轴随机移动中点这一基本思想。开始时是一条水平线段。确定其中点，使其向上或向下移动一段随机地确定的距离，然后把由此产生的两个线段再分，并使其各自中点也按此规则移动。用类似于再分三角形的方法可把这一过程不断地进行下去。

程序 MOUNTAIN 有两个数组，叫做 points 和 lines。其作用是保持计算机屏幕上的山的轮廓。每个数组分别有两列和足够多的行(比如说 2 048 行)以方便地调整屏幕分辨率。points 的两列是坐标值，而 lines 的两列则是下标。每条线段定义为数组 points 中表明该线段终点坐标的一对位置。观察一个普通的多边形通过一连串的再分后形成山的轮廓这一过程是非常有趣的，所以程序 MOUNTAIN 使每一次图案的形成都处于用户的控制下。在一次主循环结束时，程序询问用户是否需要另一次迭代，如果回答是肯定的，那么执行会再返回此程序的开头。

主循环的作用是把当前的点与线段的集合变成大 1 倍的新集



合。为实现这一点，它一次一行地对数组 lines 进行扫描，查寻其对应点的下标并从数组 poinlts 中检索出它们的坐标。在已知某一给定线段的两个端点坐标后，程序就可以计算出该线段的中点坐标，同时随机地改变 y 坐标的值。下面所列出的算法过程为程序的编制提供了充分的基础，其中变量 j 和 k 是指数组 points 和 lines 中当前正保持着再分的最新结果的那些“行”。变量 pts 和 lns 记录在进入主循环之前构成山的点和线段的数目。开始时 j 等于 pts，k 等于 lns。下标 i 从 1 到 lns。

```
j←j+1
k←k+1
a←lines(i,1)
b←lines(i,2)
x1←points(a,1)
y1←points(a,2)
x2←points(b,1)
y2←points(b,2)
points(j,1)←(1+x2)/2
points(j,2)←(y1+y2)/2+random(range)
lines(i,2)←j
lines(k,1)←j
lines(k,2)←b
```

MOU NTAIN 程序的这一部分在很大程度上是不言自明的。当第 j 点的坐标计算出来后，下标 j 就被存贮起来作为第 i 条线段的第二个点和第 k 条线段的第一个点。第 i 条线段的第一个点与其原来的一样，而第 k 条线段的第二个点与第 i 条线段原来的第二个点，即带有下标 b 的那个点相同。当循环最终计算后，pts 和 lns 必须分别复置为 j 和 k 的最新值。变量 range 是在程序的开头由用户确定

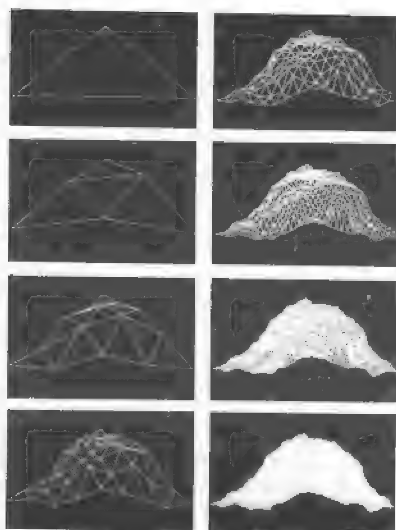


图　对三角形的反复分割产生了山的图象



各再分点在垂直方向上随机移动量的最大值。每次循环结束时,该变量就要除以 2,使得这一随机移动量与线段尺寸成比例地减小。函数 random(range)用于表示在 0 和变量 range 的当前值之间所选择一个随机数。

如果 Frodo 身后的那些山峰是令人难忘的,那么,他周围的树木和植物就更是令人难忘。它们既逼真又奇特。之所以逼真,是因为它们有与真实植物一样的分枝,而之所以奇特是因为它们不是常见的物种。大概是图形设计者有太多的参数可以任他使用,因此他禁不住要创造一些新的植物种类。

这些新的植物种类被叫做"嫁接"(graftal)植物,因为它们是在图形(graph)的基础上形成的,且有内在的的分数维性质。这里所谓的"内在分数维性质",指的是用于生成植物图案的基本拓扑特征的规规则可以(但实际上没有)应用于屏幕分辨率的极限。简言之,植物的细枝条不会无限地回归成更小的枝条。一旦作为植物的基础的图形发展起来,计算机就能用大小、颜色、厚度、质地等解释植物的图形,从而把它变换成无数的令人信服的植物种类。

某一给定植物所据以形成的图形是由 L 系统产生,这种系统是丹麦生物学家和数学家 Aristid Lindenmeyer 在 1968 年提出的一种语法类别。一个 L 系统实际就是一套用于从旧的字符串中推导新的字符串的规则。例如,根据下列规则,用数字 0 和 1 以及符号[和]能够生成一系列复杂的植物:

0→1 [0] 1 [0] 0

1→11

[→[

]→]

为了弄清如何应用这些规则,我们从由单个的符号 0 组成的字符串开始。将箭头左边的每一个符号都用与其对应的右边的符号来



代替，就可以一个接一个地得到下列的字符串：

0 1 [0] 1 [0] 0 11 [1 [0] 1 [0] 0] 11 [1 [0] 1 [0] 0] 1 [0] 1 [0] 0

把每个数字(0 或 1)当作一条线段，每个括号当作一个分支点，就可把这样的字符串变换成树一样的图形。0 和 1 所代表线段的长度相等，其区别在于 0 线段的外端上要加一片叶子，而 1 线段上则什么也不加。

例如字符串 1 [0] 1 [0] 0 的茎是由三个不在括号内的符号组成的。最下面的是 1 线段，中间也是 1 线段，顶部则是 0 线段。两根枝条(每根均是一条 0 线段)从茎上长出来。第一根枝条长在第一条 1 线段上，第二根枝条则长在第二条 1 线段上。读者可以试画一下树茎最初几次生成的图案。为了使植物更逼真，对这个模型可以加上另外一些解释性的规则。例如，对于任何给定的茎(不管它是否主茎)，都可以使枝条轮流地从左右两侧长出。

一个叫 PLANT 的由两部分组成的程序产生上述序列中的第 n 个字符串，然后把它表示成一个线段图。在该程序的第一阶段，PLANT 将它所生成的字符串保存在被称为 String A 及 String B 的两个符号数组中。每一代植物图形轮流地占据两个数组中的一个，即某一数组中所存贮的那一代是由另一数组中所存贮的上一代得来的。也不一定非要在数组中存贮符号。只要程序的代换过程是正确的，数字 0，1，2 和 3 也完全可以。

L 系统规则在条件语句中体现出来。例如可以采用下面这段算法编码把 String A 的第 i 位上的 1 个 0 变成 String B 中的九个新的符号：

如果 string A(i)＝0，那么

string B(j)←1

string B(j+1)←2



string B(j+2)←0

string B(j+3)←3

string B(j+4)←1

string B(j+5)←2

string B(j+6)←0

string B(j+7)←3

string B(j+8)←0

j←j+9

这里0和1代表它们自己，而2和3分别代表[和]，如果 string A 的第i个符号是0，那么，程序把序列1,2,0,3,1,2,0,3,0插入数组 string B 中以下标j(即数组 string B 的尚未填入符号的第一个位置)开头的九个连续位置上。程序 PLANT 的第一阶段中的一个单循环就含有四个上述的条件语句，每个语句相应于可能遇到的一个符号。循环用下标j来指出当前这一代中正被处理的那个符号。循环执行的次数依用户的愿望而定。在每一次生成后，程序 PLANT 会询问用户是否希望另一个更长的字符串。

PLANT 的第二个阶段(即绘图阶段)把第一个阶段产生的字符串变换成一个图形。它循环地执行这一过程：只要左括号(或2)没有出现，它就在一个给定方向上绘出一系列线段。当碰到某一对括号中的左括号时，程序就在一个新的方向(从前一个方向反时针转45°)上绘出后面的线段。当对应的右括号出现后，这一过程就终止。这时画出一片叶子，它的形状和颜色都留给读者去想象。第二个左括号的出现使该程序又重复进行。只是现在的方向是顺时针45°。其他的工作都是自动进行的。

PLANT 用了一个随被绘出的植物的复杂性而定的比例因数。例如，第n代植物的高度大约为 2^n 条线段，如果屏幕的高是200个像元，那么每根线段就必须短于 $200/2^n$。雄心勃勃的读者们无疑会



尝试生成语法、枝条角度及叶片形状等方面的新花样。如果具有这些新花样的图案在同一屏幕上生成，植物和树木的风景就会出现了(当然不是很逼真的)。

Pixar绘图计算机的心脏是一个有24兆字节，2 000×2 000像元的存贮器，其分辨率对大多数应用是足够的。此外，每个像元由48个存贮位表示，足够存贮色采和透明度方面的信息。Pixar绘图计算机的大容量存贮器由四个高速并行完全可编程序的处理机操纵。它们每秒钟能执行约4 000万条指令，其速度比普通的计算机大几个数量级。显示装置与存贮器间的数据交换速度可达每秒4.8亿个字节。

Pixar绘图计算机预定用于医学成像、遥感、工程设计及动画片制作这些领域中。也许还会用来制作我在本文开头所描述的假想电影。

37. 掀起滚滚波涛的细胞自动机

一种以多值方格网为基础的计算机模型——细胞自动机——已经如滚滚波涛般席卷了物理学、数学和其他一些科学领域。现在,又有一种新的细胞自动机真正生成了它自己的波。这种细胞自动机的设计者们称其为"大杂烩"机(hodgepodge machine),它能模拟化学反应,其模拟的精确性是其他模型难以与之相匹敌的。

大杂烩机所模拟的化学反应发生于易激发的化学物质中,即发生于两种或两种以上在催化剂作用下能分解并重新化合的化合物中。如果反应物的化学状态具有不同的颜色,我们就能看见一些波状结构沿着简单或复杂的边界传播,永不休止地力求达到一种难以捉摸的平衡状态。

细胞自动机可以看作是由正方形单元构成的一个无穷方格网,各单元随一台假想时钟的滴答声而不断发生变化。在时钟滴答一下时,每个单元都处于一系列有限状态中的某一种状态。一个单元在时刻 $t+1$ 时的状态按一种相当简单的规律取决于其周围单元在前一时刻(即时刻 t)的状态。这种依赖关系用一套规则来表示,它对方格网中的所有单元都同样适用。如果给方格网的各单元状态规定



一个任意的初始构型,并在时钟每滴答一下时都运用上述规则,这个初始构型就将随时间而不断变化。在有些情况下细胞自动机将会产生出一些非常奇特的图形,看到这些图形的人会情不自禁地相信,只要有合适的初始图案,细胞自动机就能产生出具有自我组织、生长及繁殖等能力的东西——言以蔽之,能产生出某种"活的"东西。

读者们最熟悉的细胞自动机大概就是 John Horton Conway 在 60 年代所发明的著名的"生命游戏"了。在生命游戏中,每个单元仅有两种可能的状态:"活"与"死"。生命游戏的规则非常简单。如果某一单元在时刻 t 是死的,则只要它的邻居中恰有 3 个在时刻 t 是活的,它在时刻 t+1 就会变成活的。如果某一单元在时刻 t 是活的,而它的邻居中在时刻 t 时活单元的个数少于 2 个或多于 3 个,则它在时刻 t+1 就会死去。凭借这两条规则,生命游戏细胞自动机就足以显示出一系列令人眼花缭乱的行为来,而且它的行为完全取决于游戏开始时死单元与活单元的组成图形。

大杂烩机不是一台,而是许多台细胞自动机。只要规定若干参数(如状态数),即选定了一种特定的细胞自动机。如果状态数为 n+1,则一个单元的每种可能状态均可用 0 与 n 之间的一个数来表示。在大杂烩机中,处于状态 0 的单元被称为"健全"的单元,而处于状态 n 的单元被称为"有病"的单元。0 与 n 之间的所有状态都受了不同程度的"感染",依状态数的大小而定。一个单元的状态越接近 n,则所受的感染就越严重。根据各单元情况的不同(即究竟是属于健全、感染还是有病),大杂烩机有选择地对每一单元运用三条规则中的某一条(见彩图 12)。

如果一个单元是健全的(即处于状态 0),那么当时钟下一次发生滴嗒声时,它的新状态取决于其邻居中现时受感染的单元数 A 与有病的单元数 B,此外还取决于两个参数(分别记为 K1 与 K2)。具体地说,这个单元在时刻 t+1 的状态由下式给定:



〔A/K1〕+〔B/K2〕

式中的方括号表示舍去括号内得数的小数部分。例如,如果 A/K1 的结果为 2.725,则加上方括号后就把这个数简化为 2。如果该式得出的结果为零,则该单元自然就还是健全的单元(至少暂时还是健全的)。

如果一个单元是受感染的,则它的情况通常是越来越糟。在时刻 t+1 它的状态为两个数之和,即该单元的邻居在时刻 t 的感染程度与一个不变量 g 之和。不变量 g 决定了感染在单元中传播的快慢。感染程度等于该单元及其邻居的状态数之和 S 除以受感染的邻居单元数 A。这样,在时刻 t 处于受感染状态的一单元在时刻 t+1 的状态由下式给出:

〔S/A〕+g

不过,这单元的状态数不能超过 n。如果上式给出的结果大于 n,则该单元的状态数就取为 n。

最后,如果一个细胞在时刻 t 是有病的(即处于状态 n),则在时刻 t+1 它就神奇地变成健全的单元(即其状态变成 0)。

除了上达规则外,对一个单元的"邻居"究竟包括哪些单元也必须作出规定。以前本专栏所提到的细胞自动机中,曾使用过两种类型的邻居,即 Von Neumann 邻居与 Moore 邻居。某一单元的 Von Neumann 邻居由该单元四侧的 4 个相邻单元构成;而 Moore 邻居则除了包括 Neumann 邻居外,还包括仅与该单元的四个顶点相接触的那 4 个单元。因此一个单元的 Moore 邻居总共有 8 个单元。给定了上述三条规则及邻居的定义后,大杂烩机就由下述四个参数的值所完全确定:n(即状态数减 1),K1 和 K2(健全单元的权),以及 g(感染速度)。

在一个 20×20 方格网上采用 Von Neumann 邻居所作的一次示范实验显示了大杂烩机的典型行为(方格网边缘上的单元仍然服从

上述规则,只是其邻居中的单元数少一些而已)。参数 n、K1 和 K2 的值分别固定为 100、2 与 3。在 g 取不同的值时,大杂烩机显示了 4 种类型的行为。在一次典型的试运行中,为这个 20×20 方格网中的 400 个单元选择了一个随机的初始状态分布图,规定了 g 的值,然后让大杂烩机运行了 10 000 个计算周期。由于一维数据比二维图像容易分析,因此仅把每一周期中受感染的单元数记下来,以便将此结果用图表示出来(见图)。

当 g 值较低时,大杂烩机没有发生多少值得注意的情况。网格中各单元的状态经过几次变化后便趋于平静,最终都成为清一色的永远健全的单元,这当然是十分令人厌烦的。但是,随着 g 值的增大,便开始出现了一些新奇的情况。首先出现的一种情况是,多数单元受到感染并一直保持受感染的状态,不过偶尔也出现一些呈不规则分布的健全单元。这种类型的行为称为类型 1。

下一种行为类型称为类型 2。类型 2 的特点是出现一系列通常是规则的被感染“高原”,持续约 30 个计算周期,其间出现大量的健全单元(有时几乎所有 400 个单元都变成了健全的,但随后就发生了新的一轮感染)。当 g 继续增加时,类型 3 的行为就出现了,其标志是受感染的单元几乎完全充满方格网和几乎完全消失这两种情况非常有规则地交替出现,大约每 20 个计算周期交替一次。最后出现的是类型 4 的行为:在大杂烩机启动后的几个计算周期之内,受感染单元的数目就开始围绕一个大约为 75% 的饱和值而有规律地上下波动。

上面四种类型的行为随着 g 的逐步增大而依次出现,但其出现也有某些重叠的情况。当 g 取转变值时,大杂烩机的运行有时产生这一种行为,有时却产生另一种行为。在有些情况下甚至发现在一次运行中都会出现各类行为间发生转换的现象。

这四种行为标志着某些特定类型的波状图案的出现(见彩图 12



中的彩色图案)。这些彩图中的方格网大小从 100×100 单元到 500×500 单元不等。相应于类型 1 行为的波在穿过一段较短的距离后就开始消失。类型 2 的波以圆形条带的形式向外扩展,条带宽度的变化相当大。类型 3 的波也呈同样的圆形条带形式,但其形状更规则,与其曲线图中所显示的受感染单元数的有规则起落保持一致。最后,类型 4 的波呈一种由方格网中心向外扩展的螺旋状图案。同以前一样,我要求有计算机的读者用某种形式重复上述实验。读者们在观察显示屏上的波时,他们的头脑中肯定也会出现相应的思维之波。

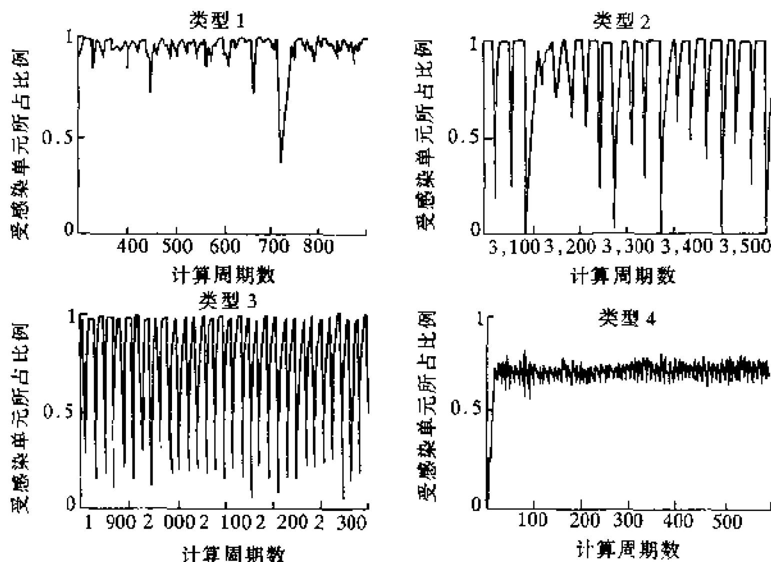


图 大杂烩机的四类行为

大杂烩机所产生的某些波状图案与许多化学系统所显示的波状图案相类似;特别是,有些图案与著名的 Belousov-Zhabotinsky 反应中所出现的波极其相像。例如,读者可比较一下彩图 13 中的计算机



生成图案的复杂花体图案与 Belousov-Zhabotinsky 反应的照片。

是什么原因造成这种相似性？这种相似性并不完全令人感到意外，人们正是有意设计了这台大杂烩机来模拟一类“多相催化反应”的特性。在这个反应中，一氧化碳和氧相结合，形成二氧化碳，同时被吸附在散布于一种多孔介质中的成千上万颗微细钯晶粒的表面上。氧化反应进行过程中所放出的热使催化剂的状态发生变化。晶粒的急剧相变释放出其表面上所附吸的一氧化碳；于是催化体冷却，反应又重新开始。

大杂烩机已被证明不仅能模拟上述这种反应，而且也能很好地模拟 Belousov-Zhabotinsky 反应。在 Belousov-Zhabotisky 反应中，丙二酸在一种催化剂(铯或铁)的作用下被溴酸钾所氧化。大杂烩机的方格网单元实际上就代表了催化剂微粒，而所谓“感染”则表示催化剂微粒的表面的逐步饱和。

但大杂烩机与化学反应间的比拟并非如此简单。其中有一些相当微妙的问题。例如，大杂绘机中的相邻单元可以说是通过交换感染而相互作用的。那么催化剂微粒是如何交换反应性的呢？至少在一氧化碳发生氧化反应的这种情形里，参与反应的催化单元通过两种基本的机制对其邻居产生影响。某一单元可以因从一个活性更强的相邻单元传递来的热量而增强其反应性，也可因从一个活性较弱的相邻单元扩散出一氧化碳而增强其反应性。

大杂烩机的相邻单元间的相互作用使它们能协调其活动以达到彼此同步。在经过开始的一段混乱时期(即杂乱阶段)后，就出现了体现这种同步性的图案。真实的化学反应的情况大概也是如此。那么大杂烩机是否就解释了它所模拟的反应中的激发波图案呢(见彩图 13)？

有些读者可能不假思索就立即声称“当然是的！”，并指着那些图案作为证据。但是也有人认为细胞自动机存在于万物之中。大杂烩



机毫无疑问有重要意义，但是它的发现者的态度则更为重要。尽管大杂烩机能很好地模拟 Belousov-Zhabotinsky 反应，但不能因此就声称化学是可用单元自动机来解释的科学。相反，应当把单元自动机看作是一种近似工具，是离散形式的偏微分方程。

大杂绘机的创造者们希望证明。一个按照某些简单规则进行局部相互作用的化学振荡器阵列最终将产生波。可能的波的形态数也许只有很少几种，但在三维空间中它们将变得复杂得多。由于在实验室的玻璃仪器中很难看见三维波前，计算机模拟可以告诉化学家们应当寻找些什么。科学研究的诀窍在于巧妙地运用模型，而不是被模型所左右。

想要建造自己的大杂烩机的读者们现在已经得到了有关如何着手进行此事的相当多的提示。首先必须说明一个具有适当大小的数组，并把它纳入一个大循环中，这个大循环按前述三条规则不断地更新数组，然后把数组显示出来以启发杂烩机的建造者。每个数组元素都必须包括某一单元的状态数。但在计算更新后的数组时，必须把计算结果暂时存贮在另一个数组中，直到计算完成。然后用一个简单的二重循环就可完成用更新后的数组代替原先的数组的工作。

数组的更新也是用一个二重循环完成的。用两个下标变量（比如 i 和 j）对方格网的单元进行计数。对于由坐标(i,j)给定的每一单元，程序（我们可否不称它为“大杂烩”呢?）根据一对“if”语句来判定该单元是健全的还是受感染的。如果是健全的，则根据第一个公式进行计算。如果它是受感染的，则根据第二个公式进行计算。在上述两种情况下，都必须对其相邻单元的状态进行检查。如果该单元既非健全的又非受感染的，那么显然就是有病的单元。在下一计算周期中它将变成健全的单元。

38. 精美的 Mandelbrot 数集及 Julia 数集

Mandelbrot 数集是通俗数学领域内的一颗最引人注目的新星。它既神秘莫测又非常美丽。的确,Mandelbrot 数集的美丽图案只是它的深刻内涵的外在表现。不经意的观察者可以看到数集的边缘布满了丰富多彩的极细的丝和花体图案,却不会想到这些图案体现了各种类型的混沌和有序(见彩图 14)。这个数集与动态系统的稳定性及混沌有很重要的联系。这种联系是通过与 Mandelbrot 数集有密切关系的称为 Julia 的一类数集建立的。Julia 数集得名于法国数学家 Goston Julia 的名字。一个 Julia 数集对应于 Mandelbrot 数集内(或外)的每个点。这种数集具有一种内在的分数维性质,所以看上去非常美丽(见彩图 15)。在介绍 Julia 数集之前,我们最好还是先回顾一下以 Benoit B. Mandelbrot 的名字命名的 Mandelbrot 数集。

Mandelbrot 数集处于复平面上。所谓复平面就是用于表示复数的一个普通平面。更明确地说,复平面上的每个点用一个形为 $a+bi$ 的数代表。我们不妨把 a 和 b 想象成那个点的坐标。a 坐标被称作 $a+bi$ 这个数的实部,b 坐标被称作虚部,i 的作用是把实部与虚部区分开来。这类复数之间可以相加,其法则是把复数的两个部分分别



相加,结果就得到一个新的复数。复数也可按照高中代数的法则相乘:

$$\begin{array}{r} 3+7i \\ \times 2-4i \\ \hline 6+14i \\ -12i-28i^2 \\ \hline 6+2i-28i^2 \end{array}$$

为了把得数表示为一个复数,需要用虚数的一个重要性质(即 $i^2=-1$)对 $28i^2$ 进行化简。因此 $6+2i-28i^2$ 就成了 $34+2i$。现在,我们来讨论一个关键的公式了,正是这个公式为我们打开了 Mandelbrot 数集的大门,给我们引出了 Julia 数集,并通过一条特殊的途径把有序状态变成了混沌。这个公式就是:

$$z \leftarrow z^2 + c$$

其中 z 和 c 都是复数,由实部和虚部组成。按照复数的乘法和加法法则,可以对 z 进行平方,然后加上 c。对这个公式进行迭代运算,即把上一个 z 值的输出作为下一个 z 值的输入代入公式进行反复运算,这个公式就变得生动有趣了。迭代的结果是得到一连串复数,它们在复平面上跳着一种奇妙的快步舞。每作一次迭代,新的复数 z 就离前面的一个 z 一定的距离。在 Mandelbrot 数集的计算中,距离这个概念很关键。

我喜欢把这一连串的由迭代公式计算出的复数(也即复平面上的点)想象成是最初那个点在漫游徘徊。那么,这个点是渴望不受限制地在复平面上跳舞,一直跳到无穷远的地方去吗?有的复数有这种运气,但其他复数却被永远限制在平面上某个有复杂形状的区域内。我们不妨把这些复数称作"囚徒",它们的监狱(即它们被限制在其内的那些区域)有分数维的"墙"。

前面我已经暗示了上述公式的迭代过程就是一遍又一遍地作反



复的计算。但是怎样选择 c 的值和最初的 z 值呢？一个办法是将 z 的初始值永远定为 0，对 c 却另选一个不为 0 的数。"囚徒"会逃出来吗？为回答这一问题，我们使 c 在复平面的某个部分上有规律地变化，用不同 c 值反复进行迭代。如果"囚徒"逃出来了，就把 c 作成白色；如果没有逃出来，就把 c 作成黑色，于是监狱的墙就呈现出了 Mandelbrot 数集的形状。如果给逃掉的点标示出的颜色不是白色，而是随逃跑的速度而变化的彩色，那么就会出现更美丽的图案来。

按照上述规则，z 的起始值为 0，即 0+0i。如果采用另一个固定的复数（比如 z=3.5+6i）为起始值，那又会出现怎样的情形呢？得出的数集会是另一种形状吗？实际上，结果总是得到一种变形的 Mandebrot 数集。我们宁愿用典范的 Mandelbrot 数集。

还有另一种规则，它刚好与上述规则相反，即把 c 值固定，而 z 则作为原始点。这样得出的数集看上去与 Mandelbrot 数集大不相同。这种数集（更确切地说是 Mandelbrot 数集的边界）叫做 Julia 数集。这类数集多得不计其数。只要给那个迭代公式选定任一个 c 值，就能产生出一个与众不同的 Julia 数集，其中关满了"囚徒"。

现在我来介绍一个定理。对于 c 的某些值，Julia 数集很明显是连通的，也就是形成单独一个区域。但对另一些 c 值，Julia 数集却不是连通的。那些打算编程序来显示 Julia 数集的读者可能已经注意到了这一点。是什么原因造成了这种差别呢？答案既简单又有趣：如果 c 点是从 Mandelbrot 数集内选出的，相应的 Julia 数集就是连通的；如果 c 点选出 Mandelbort 数集之外，其 Julia 数集就不连通。

我们可以通过动态图案来解释这个普遍适用的理论。从 Mandelbrot 数集内的任意一点向该数集外的另一点画一条直线 L，并想象 c 点不停地沿着 L 从 Mandelbrot 数集的内部向其边界慢慢移动。此时相应的 Julia 数集呈现出逐渐地收缩卷曲的形状。当 c 达到了 Mandelbnot 数集的边界时，Julia 数集已经收缩成了脆弱的树枝状脉



络,不再包含任何区域。一旦 c 点穿出边界,Julia 数集就猛然爆开,构成了一片迷朦的分数维图形。

会编程序的读者可以用他们自己选择的程序设计语言写出某些基本算法来研究 Mandelbrot 数集和 Julia 数集。这些算法都有核心迭代过程。这个迭代过程与某一定理密切有关。如果迭代值 z 的大小达到 2,那么 z 肯定趋向无穷大,再也不能返回。在大多数情况下,这一点使我们能把"囚徒"与"逃犯"区别开来。算法允许迭代值在经过 100 次迭代后达到 2。但即使经过 100 次迭代后,仍有少数的"逃犯"不能达到 2 这个值,所以这种辨别方法不是百分之百的精确。当然,为了获得更精确的图像,也可以作 1 000 次迭代。但这样做(即使是使用高速计算机)太费时间了。

复数 a + bi 的绝对值就是 a^2+b^2 的平方根,换句话说,也就是该点到复数 0 的距离。核心算法如下:

n←0

当 n<100 且 mag(z)<2 时

$z \leftarrow z^2 + c$

n←n + 1

标示出当前点的颜色

这里,下标变量 n 从 0 开始。控制迭代过程的是一个当循环;每迭代一次,n 就增加 1。只要 n 还没有达到 100,z 的绝对值没有达到 2,这个当循环就不断地按那个基本公式进行迭代。如果有一个条件达到了,此算法就退出那个循环。作色的方式我留给读者自己去决定。当然应该按某种简单的方法依照 n 的大小来标志颜色,n 的大小反映了一个点"逃跑"的快慢。读者还必须记住:图像上的点是一对屏幕坐标,这对坐标与要标在它上面的复数的坐标是不同的。

读者们编写的程序中还必须包括计算 z 的绝对值的一段算法。在上面的算法里 z 的绝对值表示为 mag(z)。的确,大多数编程语言



没有用于复数的手段,所以 z 必须分成两个部分,比如 x(表示实部)和 y(表示虚部)。c 也应如此,比如可分成 a 和 b 两部分。因此,下面的算法更接近于工作程序:

n←0

当 $n<100$ 且 $x^2+y^2<4$

xx←x^2-y^2-a

y←$2xy+b$

x←xx

n←n+1

标示出当前点的颜色

聪明的读者会注意到上面这个基本的迭代程序用了一点小小的技巧,即把 x^2+y^2 和 4 比较大小,而不是把 x^2+y^2 的平方根与 2 比较大小。这样结果不变,但却可以避免连续地求平方根,因为求平方根是一种比较费时间的运算。当新的 y 值正在被计算的时候,变量 xx 暂时保存那个刚计算出的 x。旧的 x 值被保留下来进行下面的计算,然后再用 xx 取代。

一个被称之为 MANDELZOOM 的程序把基本算法置于一个循环中,此循环不是让 a 和 b 而是直接让 c 作系统的改变。比如,如果图象的大小为 100×100 个像素(即格点),它就必须是下面这样一个双重循环:

gap←side/100

a←acorner

对于 j←1 至 100

a←a+gap

b←bcorner

对于 k←1 至 100

b←b+gap



x←0

y←0

[基本算法]

在执行过程到达这些指令之前，MANDELZOOM 允许用户输入一个复数，它位于用户所要考虑的正方形区域的一个顶点上，坐标是 acorner 与 bcorner，即 a 和 b 在该正方形区域上所取的最小值。这个正方形的大小由程序用户确定，此程序正是由这个正方形而得名的。它是一个供我们窥看 Mandelbrot 数集的奥秘的窗口。这个窗口可以作得很小很小，其结果就使我们把目光移向它所圈着的那个范围。程序 MANDELZOOM 还要求用户输入 side 的值，即复平面上那个正方形的边长。然后，算法计算出相继的 c 值之间的间隔，从而得出了 a 和 b 的适当增量。

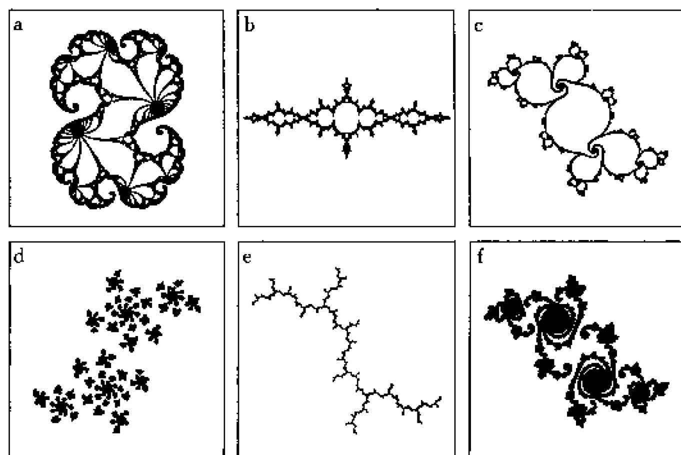


图　6 种 Julia 数集，其中有些是连通的(a、b、c 与 e)，其余是不连通的(d 与 f)

下标 j 和 k 的值不参加双重循环内的任何计算，所以可以把下标改成某种更适用的形式。例如，j 和 k 可以不是从 1 变到 100，而

是各自经过100个连续的坐标点。当基本算法决定了迭代值z的颜色后，坐标点(j,k)就被标示出这种颜色。

对于产生出Julia数集的图像的程序，我觉得还是只有沿用前面的办法把它称作JULIAZOOM，尽管我并不喜欢这个名字。通过这个程序，我们也能够把镜头对准某个数集，在很高的放大倍数下观察它。JULIAZOOM使用了与MANDELZOOM相同的核心算法，只是表面上略有不同。

JULIAZOOM要求用户首先输入xcorner，ycorner和side，同时要求用户以变量a和b的形式给出c的值。接下来这个程序应用了与MANDELZOOM相同的双重循环，但它们之间有某些重要的不同之处：

```
gap←side/100
x←xcorner
对于 j←1 至 100
x←x + gap
y←ycorner
对于 k←1 至 100
y←y + gap
[基本算法]
```

基本算法根据n的值来对屏幕上的点作色。有些最有实际意义的图案就产生于最简单的赋色方法。在彩色监视器上，根据下面的方法作出三种颜色所得的图案的效果也是非常好的：把n为0～10的点标上第一种颜色，n为11～20的点标上第二种颜色，n为21～30的点标上第三种颜色。然后又把n为30～40的点标上第一种颜色，如此类推。在单色监视器上，以10为区间按n的大小交替使用两种色彩，就可获得黑-白色(或绿-黄色)效果。

读者们有了MANDELZOOM和JULIAZOOM这两个程序，就



能够在自己的计算机上探索这些极为壮观而有意义的分数维数集，也能够游览数集附近的复平面，或者用上面提到的计算机显微镜观察复平面上某些特定的部分。当接近由计算机的算法精度所确定的显微镜分辨率的极限时，这两种数集的图形都显露出迷人的细部来。下面的坐标范围划出了这两类数集的区域，可为游览分数维的无穷小世界的人们提供指导：

对于 Julia 数集，x 和 y 的范围为 -1.8 至 $+1.8$

对 Mandelbrot 数集：x 的范围为 -2.25 至 $+1.75$

y 的范围为 -1.8 至 $+1.5$

对于钟摆和电子电路之类的动态系统，可以用一个不含复数而含实数的简单迭代公式来概括它们的动态特征，这个公式是：

$x \leftarrow r\,x(1-x)$

这个公式显然是一个二项式：作乘法运算后就出现 x 的平方。按此公式作迭代运算，其结果是简单或是奇异，取决于你所确定的 r 的值。对每个 r 值，迭代结果最后都稳定在一条轨道上，这条轨道由 x 有规律地反复取的若干个值构成。当 r 趋近 3.569 9 这个临界值时，x 就在一大群可能取的值中狂乱地振荡，无法对它作出预测。这种情况相当于上述迭代公式所代表的系统（不论是钟摆或是电子电路）在寻求稳定状态时完全陷入了困境。它以一种无法预测的方式胡乱地徘徊。这就是混沌。

本文讨论的复数迭代公式 $z \leftarrow z^2 + c$ 也有与此相似的现象产生。对于一个给定的 c 值，存在一个以上的吸引轨道，轨道的多少取决于 z 的起始值。如果 z 的起始值比较小，z 的迭代运算结果就趋向于一个特定的点；如果起始值大，运算结果就将无限变大，无穷大就是该系统的吸引子。特定的点和无穷大对复平面上的点构成了两个独立的单点吸引轨道。这两者的引力范围之间的边界，即 Julia 数集本身，其形状之卷曲折皱简直令人难以置信。这条边界也是一条轨道，



但从理论上讲它不存在吸引力。已经位于边界上的那些点在边界内无规则地跳来跳去。然而,由于计算机的数字精度可能不充许你精确地在这条边界上规定一个起始点,所以 Julia 数集不容易直接地计算出来。在迭代过程中精确度将下降,迭代值最终将不知跑到什么地方去了。

如上面讲的,每个可能的 c 值都将导致一个新的 Julia 数集的出现。从某种意义上讲 Mandelbrot 数集一举概括了所有可能的 Julia 数集。对所有可能的 c 值,它都描述了复数 0 迭代运算的结果。某些 Julia 数集的混沌区域仅仅是树状图形或者甚至是对称分布的胡椒点状的图案。读者一定记得这样的 Julia 数集相当于在 Mandelbrot 数集边界上或边界外的 c 值。

可以这样描述 Mandelbrot 数集和 Julia 数集之间的关系:Mandelbrot 数集是一本很大的书,而一个 Julia 数集只是其中的一页。根据 c 点在 Mandelbrot 数集中的位置,就能预测与之相关的 Julia 数集的外形及大小,从而得知迭代值的一般情况。预测的范围远远不止它是否是连通的。例如,如果 c 值是选自 Mandelbrot 数集的主体和它的一个"芽"间的颈部,则相应的 Julia 数集就也有它自己的颈部和芽部。把 Mandelbrot 数集比作查阅 Julia 数集的词典,表明这两者之间存在一个根本区别。Julia 数集是"自相似"的,而 Mandelbrot 数集却没有此性质(甚至于它的边界也没有这个性质)。Peitgen 说如果它有这种性质,它就不能对它的不计其数的亲戚——Julia 数集——进行编码了。

最后提一下动态系统的研究者们目前正在研究的一个潜藏的高维空间中的数集。这个数集由立方迭代公式 $z \leftarrow z^3 + c$ 得出,非常奇特。它是四维数集,其分枝沿着某些看不见的方向卷曲伸展。但它的三维的横截面是能计算出来的。

39. 生物形态、Truchet 瓷砖及分数维爆玉米花



下面将向读者们描述怎样玩三个简单而饶有趣味的游戏，这三个游戏的对象分别是生物形态、Truchet 瓷砖以及分数维爆玉米花。三维对数蜗牛的味道。

何为生物形态？生物形态就是牛津大学生物学家 Richard Dawkins 编制的计算机程序所产生的栩栩如生的生物形态。此程序在产生生物形态时，先由操作人员从一组树形图案中任选一个图形，然后程序便从这个图形出发演变出另一组图形——其中所有图形都与操作人员选择的那个图形多少有点不同。操作人员通过一次又一次的选择，最后可能得到一些十分奇特而又截然不同的有机形态。

Pickover 独立地而且几乎与 Dawkins 同时研究出了他的生物形态。但是，他的生物形态的产生方法及其外貌与 Dawkins 的完全不同。例如，Pickover 的生物形态看上去明显地具有微生物的形象。

Pickover 的生物形态栖居于复平面，即有名的 Mandelbrot 数集所在的平面。Pickover 用一种描出 Julia 数集的精美的分数维几何图形的简化方法来制作他的生物形态 Julia 数集与 Mandelbrot 数集有极为密切的关系。通过对某一特定功能(即一系列数学运算)进行

反复迭代，便可产生出一幅生物形态图。前一次迭代运算的输出是随后一次迭代的输入。

例如，图 1 的左下方是有 12 个穗的穗放射虫状的生物形态图。这一幅怪诞的显微图可通过下面的迭代公式产生：

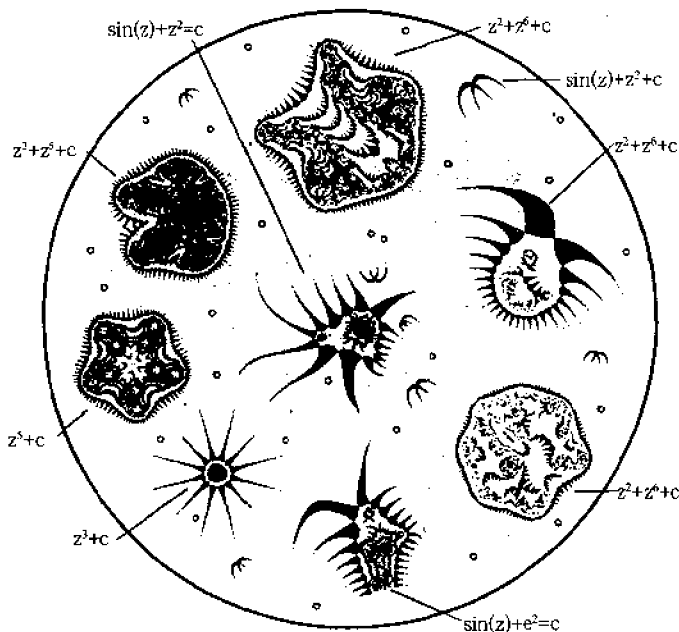


图 1　一些生物形态图的显微图像以及产生出它们的各个函数

$$Z_{n+1} \leftarrow Z_n^3 + c$$

将复变量的初始值 Z_0 取其三次幂，加上一个复常数 C，得其和 Z_1。然后再对 Z_1 进行同样的演算，得出 Z_2，以此类推。

一个复数由两部分构成，其中一部分叫实部，另一部分叫虚部。我们最好把这两部分看作是二维平面上的笛卡儿坐标。按惯例，一个复数表示为它的两个组成部分之和。例如，复数 3 + 5i 的实数部分为 3，而虚数部分为 5（i 是虚数的记号）。为了避免啰嗦，本文最后



列出复数的加法,乘法和立方法则。

要作一个生物形态图,首先必须设计出复平面上的一个矩形坐标点网格。每个点的一对坐标构成了上面介绍的迭代运算的一个初始值 Z_0 的虚数部分和实数部分。每个点同时还被指定给计算机显示屏上的一个像素。某个像素作成白色或是黑色,取决于对最终得出的 Z 值的实部与虚部的"大小"进行一个简单检验的结果。

图 1 中的全部生物形态都是在一个以复平面的原点为中心的 20×20 的正方形内发现的。有耐心的读者还能够发现其他多少种不同的形态呢? 下面我为读者们提供一个必要的观察手段,即根据 Pickover 的基本算法编制出的程序 BIOMORPH。下面的这一算法将产生出放射虫:

$c \leftarrow 0.5+0.0i$

对于 $j \leftarrow 1$ 至 100

对于 $k \leftarrow 1$ 至 10

计算 Z_0

$Z \leftarrow Z_0$

对于 $n \leftarrow 1$ 至 10

$Z - Z^3 + C$

如果 $|Z| > 10$

则退出循环

如果|实部(z)|或|虚部

(z)|<10

则标示(j,k)为黑色

反之,标示(j,k)为白色

指令"计算 Z_0"看来简单,实际上并不简单。它要求把每对像素坐标(i,k)都变换成一个复数,这就需要用 j 值的个数和 k 值的个数分别去除复平面上某个区域的长度和宽度,得出的商作为一个增量,



Z_0 的实部和虚部在此算法的每次循环中按此增量有规则地增大。

例如,放射虫实际是处于复平面上一个正方形的“窗口”之内,这个“窗口”的边界由下列 Z_0 的实部和虚部范围所确定:

$-0.5<$ 实部$(Z_0)<1.5$

$-1.5<$ 虚部$(Z_0)<1.5$

因为 j 和 k 都是从 1 变到 100,所以 Z_0 的实部和虚部有规律地每次增加 0.03。因此,上面给出的这个 BIOMORPH 程序还必须依靠下面这类语句计算出 Z_0 的 10 000 个值,且对每个 Z_0 值进行迭代和检验:

实部$(Z_0)\leftarrow -1.5+0.3j$

虚部$(Z_0)\leftarrow -1.5+0.3k$

在进行头 10 次迭代的最内层循环时,必须不断检查 Z 的大小及其实部和虚部的大小。一个复数的大小就是该复数的实部和虚部的平方和的平方根。而其实部和虚部的大小则分别为其各自的绝对值。如果 Z 或其实部或虚部的大小大于 10,程序必须立即退出循环(即使还没有进行到第十次迭代),并对 Z 的实部及虚部值的大小再次进行核查(在把 Z 的大小与 10 作比较时,较简单的作法是把它的实部及虚部的平方和与 100 作比较,而不是将平方和开方后与 10 作比较。两种计算的结果是相同的)。

不管进行了多少次迭代,如果最后一个 z 的实部或虚部中有一个的值小于 10,坐标点(j,k)的像素被着上黑色;反之着白色。

对大多数个人计算机而言,其坐标为 1 到 100 的像素都是位于显示屏的一角上。要使图像处于显示屏中心,必须改变 j 和 k 的起始值和终值。例如,j 和 k 可能要从 50 变到 150,而不是从 1 变到 100。

程序 BIOMORPH 的所有其他细节有待富于探索精神的读者们来设计。这里要提醒读者们注意,Z、Z_0 和 c 都是复数,必须按复数



的加法和乘法规则进行运算，同时，在此算法中所有涉及 Z、Z_0 和 C 的赋值语句，在用普通计算机语言编写时都代表两个赋值语句——一句用于实部而另一句用于虚部。

在 Pickover 的"显微镜"下可以看见的其他几种生物形态是如何产生出来的呢？实际上，它们也可用程序 BIOMORPH 来产生，只不过必须用另外一个不同的迭代函数作为算法的核心。换句话说，就是要用另外一个函数来取代上述程序中的 $Z^3 + C$。

编写 BIOMORPH 程序的读者可能很想在复平面的那个 20×20 的中心区域内"钓到"别的什么玩意。为此，最简单的办法是先对整个区域进行扫描，然后集中注意力检查初步扫描中所发现的生物形态的细部。通常可用任一个很小的 C 值作为"钓饵"。

有些能够产生生物形态图的函数利用了一种奇特的方幂，即 Z 的 Z 次方(Z)，其他一些函数则利用了三角运算。编程者具备了这些知识便能够编写出产生形形色色的原生动物形态的 BIOMORPH 程序。

生物形态产生于一个偶然的场合。最初 Pickover 编写了一个用于探索各种公式的分数维性质的程序，此程序中的一个错误导致了生物形态的出现。他在对 Z 的实部和虚部的大小进行条件检验时偶然用一个"OR"代替了"AND"，这个无意中的改变使得着上黑色的象素比没有发生这一改变时的黑色像素多得多。从那些生物形态上伸出的纤毛就是由这些像素组成的。

尽管这些生物形态图是碰巧产生出来的，但它们看来已经呈现出了它们自己特有的生命力。正如 Pickover 所说的那样："从某种意义上讲存在数学上的生物。它们栖息在复平面上，但我们很容易把它们想象成是大量繁殖在一滴池塘水中的显微生物。"在什么样的海洋中我们能够捕捉到更为高级的生命形式呢？Pickover 还发现，天然生物和人造生物的复杂性都来源于反复应用一些简单的动态规律



这一事实具有广泛而深刻的意义。

目前,Pickover 正在反复思索用哪些方法能够一眼就可判断出复杂数据所包含的意义。例如,在一般情况下,很难判断出用 0 和 1 表示的一整页数据是有规律的或是随机的(用 0 和 1 代表数据是数据表示的基本方法)。但是,Pickover 能够借助一种以"Truchet 瓷砖"为基础的简单绘图工具判断出数据的随机程度("Truchet 瓷砖"得名于 18 世纪法国一位博学的僧侣 Sebastien Truchet)。Pickover 把原先的 Truchet 瓷砖作了些改变。新的瓷砖图形是一个正方形,其内部有两个 1/4 圆,这两个 1/4 圆的圆心位于正方形的两个相对顶点上,且各与正方形的两条边相交于其中点上。这样得出的瓷砖只有两个不同的方位(见图 2)。

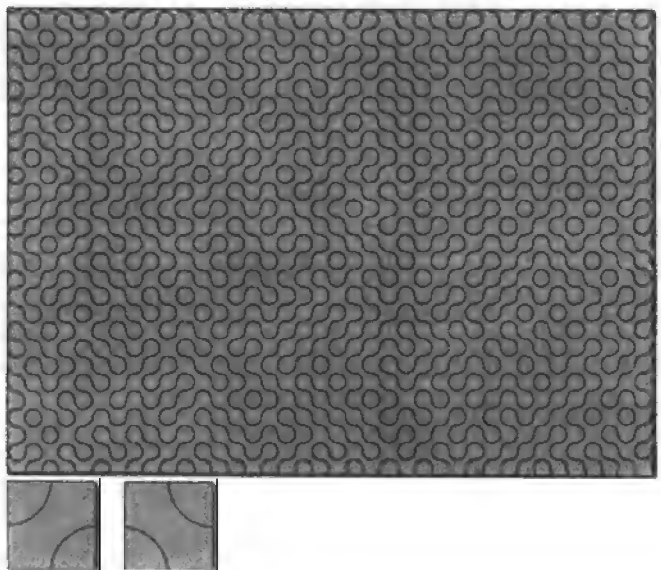


图 2　Truchet 瓷砖的两种方向(左)和一种无规则的铺砖法(上)

如果用 Pickovetr 的 Truchet 瓷砖覆盖一平坦的区域,便会出现



一些奇特的弯曲曲线。除了在这区域的边缘上外,这些曲线都不会出现断开的端头。每条曲线保证是连续不断的,因为相邻的各个正方形的边的中点(这些中点正是正方形内的1/4圆的起始点或终点)是对接的。

只要把许多瓷砖按二进制数字的顺序铺放(当二进制数字为0时瓷砖取某一方向,而当二进制数字为1时则让瓷砖取另一方向),就能把任何一个由0与1组成的阵列转换为用Truchet瓷砖铺满的区域。Pickover认为,如果这些二进制数字是随机的,那么在这个Truchet瓷砖构造中就找不出任何规律性:环状曲线和波纹完全是混乱的(尽管很好看)。

然而,只要让这些数据稍微有一点规律性,那么就可以看出曲线也具有一定的规律了。例如,如果我们作出这样一个二进制数字阵列使它的每一行中的一个0的后面有更大可能跟着另一个0而不是1,而一个1的后面有更大可能跟着另一个1而不是0的话,那么,在相应的瓷砖构造中曲线就将明显地显示出一种对角走向。读者们很可能乐意自己去探索这一现象的原因。

实际上,由随机的二进制数字产生的瓷砖花样(如图2所示的一个花样)使我们联想到种种有趣的娱乐,比如,它们是迷宫吗?读者可以试着去找到一条从瓷砖构造的上部通向底部或从瓷砖构造的一边通向另一边的通道。在沿着通道前进时,常常会碰到一些"孤岛",即由曲线自相接合而成的闭合线。任何没有碰到正方形边缘上的曲线迟早都会折回来自相接合。有些这样的曲线构成了小圆圈,有些呈哑铃状,有些则呈更为复杂的形状。建议读者把其他类型的闭合曲线进行分类,看看哪些曲线实际上可以归作一类,它们占总砖数比例是多少(取平均值)。

编写一个把二进制数字阵列转换成Truchet瓷砖的程序是很容易的。假如给定了这样一个二进制数字阵列,只要设计一个双重循

环对阵列的每一元素来进行扫描并在计算机显示屏的适当位置上画出相应的 Truchet 瓷砖就行了。关于 Truchet 瓷砖我就只讲这么多，下面我将把话题转到 Pickover 的另一个图形游戏上，即分数维爆玉米花。

分数维爆玉米花是由一对离散形式的迭代微分方程的解产生的。从上一个解的坐标中减去 y 的一个函数的值便得出后一个解的 x 坐标，同样地，从上一个解的 y 坐标中减去 x 的一个函数的值就得出后一个解的 y 坐标。这样的一个方程组叫作循环系。看来，以三角函数为基础的循环系能够产生出特别丰富多采的精采图形。

下面这个方程组表达的是在程序 POPCORN 中的交叉变量迭代过程：

$$x_{n+1} = x_n - h\sin(y_n + \tan(3y_n))$$

$$y_{n+1} = y_n - h\sin(x_n + \tan(3x_n))$$

两个迭代方程中的三角函数前面都乘上了一个较小的常量 h，这样，不管迭代过程进行多久，新的 x 或 y 的值，始终不会与其前一个值相差太大。Pickover 在他的程序中令 h 的 0.05。

正如下面这个简略的算法所示，程序 POPCORN(x_0, y_0)的2 500 对初始值的每一对都代入上述公式作 50 次迭代，每迭代一次就在计算机屏幕上绘出一个像素。

对于 j←1 至 50

对于 k←1 至 50

$x_0 \leftarrow -6 + 0.24j$

$y_0 \leftarrow -6 + 0.24k$

$x \leftarrow x_0$

$y \leftarrow y_0$

对于 n←1 至 50

$xx \leftarrow x + h\sin(y + \tan(3y))$



yy←y + hsin(x + tan(3x))

x←xx

y←yy

计算 jp 和 kp

绘出(jp,kp)

(xx 和 yy 是中间变量,用以保存一个迭代循环的 x 和 y 的现行值)

x 和 y 坐标的初始值选自于一个以 x - y 平面的原点为中心、大小为 12×12 个单位的正方形内的点。换句话说,x 的初始值是从 -6 至 +6,y 的初始值也是一样。像 BIOMORPH 一样,POPCORN 也是依靠一个嵌套循环来计算一组初始值。但在 POPCORN 中,两个循环都是从 1 运行至 50。

最后,由每对初始坐标(x_0,y_0)所产生出的 50 个点(x,y)都必须作为像素显示在计算机屏幕上。这一步是通过一个把 x 和 y 按比例增大并平移后转换成 jp 和 kp 的简单公式来完成的:

jp←4.166x + 25

kp←4.166y + 25

因此,Pickover 的分数维爆玉米花完全是由那些其坐标是用迭代运算求出的像素所组成的。读者们可以随便选出这些"爆玉米花"中的某一部位加以放大,去发现许多完全意想不到的漂亮的细节。读者们还可以像 Pickovet 那样把爆玉米花染上颜色,就是每当内循环重新开始时给像素作上不同的颜色。

复数的运算规则

两个复数相加时,将其实部与虚部分别相加。也就是说,复数 a + bi 与 c + di 之和为(a + c) + (b + d)i。

复数乘法的运算规则要稍微复杂一些。复数 a + bi 与 c + di 之积为(ac - bd) + (ad + bc)i。用这一规则可得出 a + bi 的平方公式,



即：$(a+bi)^2=(a^2-b^2)+2abi$，其实部为 a^2-b^2，虚部为 $2ab$。

进一步运用复数乘法公式可得出求 $a+bi$ 的三次方的规则。$a+bi$ 的三次方的实部为 $a(a^2-3b^2)$，而虚部为 $b(3a^2-b^2)$。

40. 蚂蚁的无尽之旅

旅馆的前厅一片混乱，到处都是一堆堆的手提箱和背包。我无数次地想到，哪怕只是考虑一下来参加这五年一度的 SPAM 世界大会，是否都是有点神经不正常。我向旅馆登记处前排起的长龙看了一眼，然后向吧台走去。参加"数学哲理化协会"（SPAM）这次会议的与会者半数都有同感。

就在我的左边，一位年约 65 岁的穿着时髦的妇女正同看来是集中体现了十几岁青年的邋遢劲的人进行着火爆的讨论。一位有着钉子般的头发的瘦瘦的妇女穿着一件 T 恤衫，其上印有"注意这个空间"的字样。一位过分激动的模糊论者正向三位持怀疑态度的构成派学者解释常规逻辑的灵活推广（这些构成派学者希望把它弄得更严密一些）。此外还有一位其貌不扬的典型人物躲在角落里拼命敲打着一台膝上式计算机的键盘。

我苦笑了一下，开始放松自己。毕竟这次出行会像我希望的那样对智力进步是大有裨益的。我向那位有着钉子般头发、穿 T 恤衫的妇女——她名叫 Louise——作了自我介绍。"我在注意看着



——”,我说道。

“嗯?”

“‘这个空间’。我看不出有什么不同寻常的地方。”

她向我做了个滑稽的表情,似乎想弄清我是不是个主张性别歧视的人。“这是说的那个方程”,她说,“当它被发现的时候。”

“方程?”

“听着,既然国会中的那些白痴已经砍掉了超级超导对撞机工程,我对迅速取得突破并不抱很大希望。”

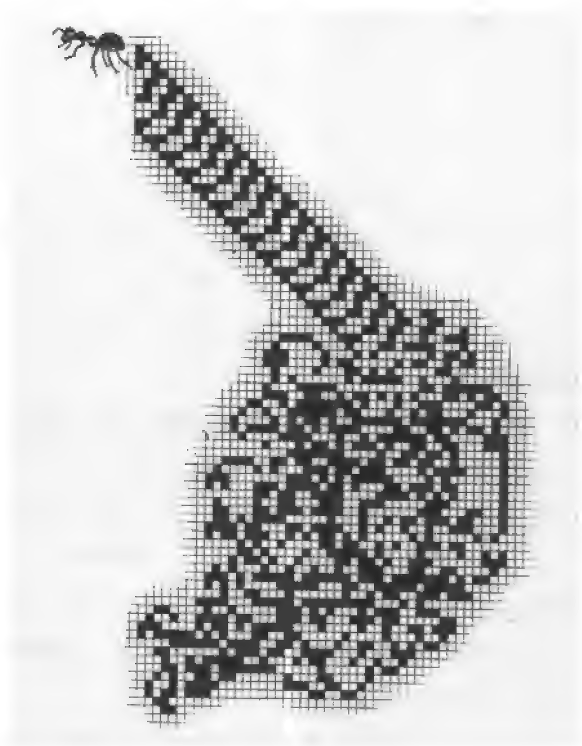


图1 在 Langton 的蚂蚁的无尽旅程中,公路建造阶段出现于大约10 000个混沌步骤之后



“呵,是那个方程”,我说。“就是万物理论。”她是个原教旨主义者;我本该看出来的。

“你可以嘲笑我”,她说。我摇了摇头,表示我压根儿没想过要嘲笑她。“我就是相信宇宙中的万物都是受一条根本法则支配的,而科学的中心任务必定是发现这一条根本法则。”

“不错。但即使存在这样一条根本法则,你怎么又认为它会是一条数学法则呢?”“‘法则’这个词本身就表明了一种只有在数学中才能获得的精确性。事实上,我们可以把数学定义为研究简单而精确的法则的逻辑推论的学问。”

“你说的‘逻辑’究竟是什么意思?”一位构成派学者问道。

“你说的‘精确’又是什么意思?”那位模糊论者——他名叫 Inez——问道。大会已在进行中。

“关键之点在于”,Louise 说,“一旦我们发现了自然界的法则,我们就能从法则推导出其余所有一切。我们得到的将是真理,而不是各种近似理论拼凑起来的大杂烩。”

“我认为你们全都弄错了讨论的层次”,我插进来说道,“数学家们证明宇宙的整个未来原则上可由其现在状态决定——从而导出时钟结构式的宇宙图景以及简单法则必定产生简单行为的观念——只有 100 年左右的时间。但是当人们开始认真地思考实际上可能发生什么情况时,他们发现了混沌——简单的法则能够产生出极其复杂的行为,而确定性的系统能够产生出随机性的行为。出于论证的需要,假定你是正确的。假定归根结底宇宙的确服从一组简单的根本法则,而我们则发现了这些法则。那么这真的有助于我们认识我们生活在其中的这个宇宙吗?

“当然如此”,Louise 回答道。“首先,它为我们奠定了一个根本的哲学基础。其次,我们人类世界的行为必定隐含于根本法则之中,因此这些法则将能解释一切现象。”



"原则上或许如此,但实际上并非如此。例如,在人类尺度的世界中,猫喜欢捉老鼠。我不会认为你所说的'根本'法则能解释这一现象,除非你可以给我拿出一个令人信服的推导过程,此过程始于你的方程,而最终必须推导出猫喜欢捉老鼠这一事实。你打算怎样做呢?"

"即使你能够执行这些计算,它们也将是庞大得令人难以对付且完全是无法理解的。是的,万物理论存在的问题就在于这类理论以错误的'解释'观念作为其出发点。所谓解释,就是从假说推导出结论的一个明确的论证过程,而不只是宣称结论隐含于前提之中这样一种模棱两可的说法。肯定也不是旨在使隐含结论变为明确结论的厚达 1 000 英里的一堆计算机打印结果。"

躲在角落里的那个人醒了过来。"我来让你们大家瞧点东西。"他把他的膝上式计算机放到了桌子上。"看看屏幕。"屏幕上出现了一个很好看的方格网。"这是只蚂蚁,对吧?"

"在哪里?"

"在中间,只是看不见而已。不过现在我要向你们显示出这只蚂蚁",他又敲击着键盘。有个什么东西在方格网上迅猛地来回飞跑,留下一道不规则的由黑白方格组成的路径。这东西在屏幕上动了 1 分钟左右,建成了一个有着奇妙图案的斜向长条,然后在屏幕的边缘处消失不见了。

"真迷人",Louise 说,"好了,正如我刚才说的关于方程的——"

"Louise",那个其貌不扬的人说道,"如果你让我解释一下你刚才看到的东西,你将会发现它同我们现在讨论的问题有密切的联系。"

"你总是这样说,Nathan,即使是对 Dougal the Dugong 游戏你也这样说。"

"且慢下结论,我将证明我的说法是有道理的。我刚才显示给你



们看的是一个称为 Langton 蚂蚁的数学系统。这是圣菲研究所的 Chris Langton 发明的一种简单得令人吃惊的单元自动机。”

“圣菲研究所？就是搞复杂性研究的那一群家伙吗？”

“正是。他们寻找复杂系统中的大尺度规律性。Langton 蚂蚁是一个简单的例子。这只蚂蚁从中央方格动身，朝着某一选定方向——如东方——爬去。它在此方向上爬过一个方格，然后看看它所进入的方格的颜色是白还是黑。如果它进了一个黑色方格，就把它涂成白色，然后向左转 90 度。如果它进入了一个白色方格，就把它涂成黑色，然后向右转 90 度。它一直按照这些简单的规则行事。你们已经看见了当我在一个全是白方格的方格网上启动这只蚂蚁后所发生的情况(见图 2)。”

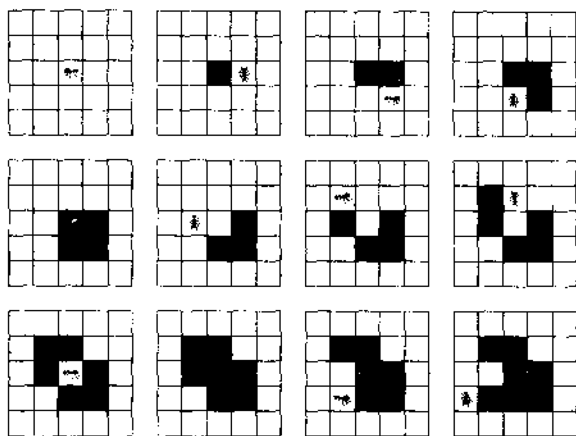


图 2　Langton 蚂蚁走出的头 12 步示于图中(为清楚起见，蚂蚁尚未到过的方格用灰色表示；在应用规则时这些方格应当作白色方格来处理)

“对于如此简单的一套规则来说，这真是复杂得令人吃惊的行为”，我指出。



“你瞧，Louise，Langton 的规则就是他的蚂蚁所在的宇宙中的万物理论。这是一个蚂蚁物质宇宙”，他以辩护的口气说完这一段话。

“看看 Langton 蚂蚁创作的这一串奇怪图形。在头 500 步中，它不停地返回中央方格，留下一系列相当对称的图案。然后，在下面的 10 000 步中，图形变得非常混沌了。突然——几乎就像是它最终拿定了主意要干什么似的——这只蚂蚁建造起麻省理工学院的 James Propp 所谓的公路（Propp 首先发现了这一现象）。它不断重复着由恰好 104 个步骤组成的一个过程，每重复一次就使它向西北方向移动两个方格，并无休止地继续下去，从而形成一条斜向的带形（见图 1）。”

“有趣极了”，我说。Louise 的脸色却似乎在问，那又怎么样呢？

“真正有趣的地方在于”，Nathan 继续说道，“它似乎最终总是要建造公路，即使你在它开始爬行之前使方格网上到处散布一些黑方格时也是如此。”

“我认为”，一位构成派学者说，“我可以让蚂蚁挨着一条无限长的黑方格带开始启动，这样它就会循着这些黑方格跑到无穷远处，只要它们的间隔合适的话。”

“对不起。我指的是有限的黑方格排布图形。不过还没有人能够证明蚂蚁总是要建造公路的。”

“如果 Langton 蚂蚁开始爬行时方格网上有一个有限的黑方格图案，那么关于蚂蚁的行为是否已证明了某个较一般的结论呢？”

“是的”，Nathan 回答说，“洛克菲勒大学的 E. G. D. Cohen 和 X. P. Kong 证明了蚂蚁的爬行路径必定是无界的。它将逸出任何有限区域（见框内文字）。”

“但这同方程有什么关系呢？”Louise 问道。

“我们知道了 Langton 蚂蚁的万物理论”，Nathan 提出，“也就是那些规则。我们建立了这些规则。但是，尽管如此，没有人能够回答



一个简单的问题:从一个由有限条黑方格构成的'环境'出发,蚂蚁总是会建造公路吗?"

"因此对于这个问题万物理论缺乏解释能力吗?"我问道:

"正是如此。它预示一切东西,但解释不了任何东西,相反,Cohen-Kong 定理则说明了路径为什么是无界的。"

"我可以发现你的论据有若干问题",Louise 指出,"首先,Cohen Kong 定理是万物理论的一个推论因此该理论至少具有一定的解释能力。其次,你的论证是以我们的无知为基础的。或许明天某个人就会提出蚂蚁总要建造公路的一个证明。"

"我同意你的看法。但是,只有在使 Cohen-Kong 推论成为明显表述后才能解释路径的无界性。你可以依赖万物理论的推论的唯一性到极端程度,但单凭这种唯一性并不会告诉你是否存在有界的路径。类似地,即使我们知道万物理论,我们对建造公路是否是蚂蚁的一种普遍行为仍将一无所知,除非某人证明它确是万物理论的一个明显推论或者否定了它。"

"我觉得",我插话说,"你仅以一种例外情况为基础就作了如此多的断言。"

"并非如此",Nathan 反驳说,"Langton 蚂蚁完全是以规则为基础的系统的典型代表。有许多推广情况,这些推广都显示了令人吃惊的行为,而更令人吃惊的是它们还显示了一些共同的模式。把一只或多只蚂蚁放进一个选定的环境中,然后观察它们的行为,将是一件富有乐趣的事情。你可以改变规则并设立不同的环境——例如采用一个六边形的方格网而不是正方形的方格网。这些操作最好在计算机上进行,因为使用简单的计算机程序就可执行规则。我还要补充一点,就是上面这些想法也有其实际的一面:它们同统计力学中关于粒子阵列的问题有关(粒子就是"蚂蚁"且它们在任一给定时刻只能处于几种状态——相当于有色方格——中的一种)。"



“粒子和蚂蚁粒子?”

“谢谢你,Inez。最近,斯坦福大学的 Greg Turk 以及佐治亚理工学院的 Leonid A. Bunimovich 和比勒费尔德大学的 S. E. Troubetzkoy 彼此独立地研究了由规则串定义的广义蚂蚁,假定方格有几种颜色而不仅是黑白两色,分别标以 0,1,2……n－1。规则串就是由 n 个符号——0 或 1——组成的一个序列。当蚂蚁离开颜色为 k 的一个方格时,就将其颜色变为 k＋1。(把 n＝(n－1)＋1 排成一圈直到零)。如果第 k 个符号 1,蚂蚁就向右转,如果第 k 个符号为 0,蚂蚁就向左转。然后蚂蚁进入另一个方格,并重复上述过程。”

“Langton 最初的规则可以表示为规则串 10。有些规则串产生的蚂蚁动态行为非常简单——例如,规则串为 1(或甚至是 111……1)的蚂蚁绕着一个 2×2 的正方形无休止地爬下去。但是,按照 Cohen-Kong 的想法,任何一种既含有 1 也含有 0 的规则串必定会产生出无界的路径。”

“为简单起见,假定开始时是一个‘干净’的方格网——也就是所有方格的颜色均为 0。蚂蚁 100 产生的图案,开始看起来很像 Langton 蚂蚁产生的图案,即最初是对称的,随后变成混沌的。但是,经过 1.5 亿步之后,它仍然是混沌的。它会不会建造公路呢? 无人知道。蚂蚁 110 要建造公路,而且只经过 150 步就到了建造公路的阶段了。此外,Longton 蚂蚁建造公路时的每一循环有 104 步,而蚂蚁 110 只需 18 步。蚂蚁 1 000 混沌得无以复加了,而蚂蚁 1 101 则始于混沌,经过 250 000 步后进入了建造公路阶段,其循环长度为 388 步。蚂蚁 1 100 不停地产生出越来越复杂的图案,这些图案无穷多次地变成双侧对称的状态(见图 3)。因此它无法建造通常意义上的任何一种公路。”

“我想看看有没有人能够提出对所有这些广义蚂蚁的行为的简单分类,或者有没有人能根据蚂蚁的规则串来预测它们的长期行



为",Nathan 宣称,"即使这些蚂蚁全都在一个干净的方格网上开始爬行。"

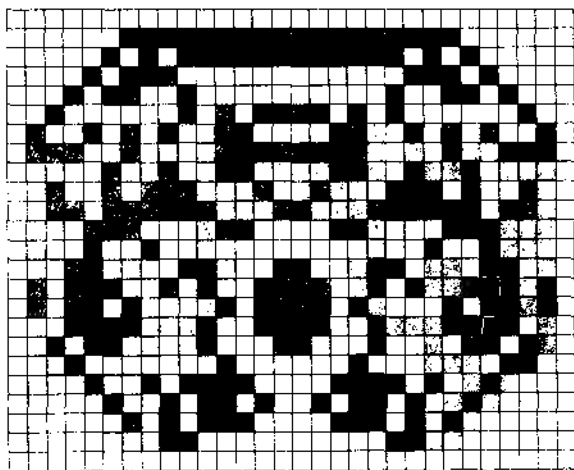


图 3 某些规则串的每一步都产生出对称图案。
本图的图案为蚂蚁 1 100 的和 16 464 步

Louise 显出一脸不高兴的样子。"是的,不过你并未证明没有人能够办到。"

"不错",我指出,"但是仅仅稍微复杂一点的转换规则就可引导到 John Horton Conway 的生命游戏之类的例子。Conway 证明了在生命游戏中存在一些构成万能图灵机(即可编程序计算机)的构型。A.M.图灵证明了图灵机的长期行为是不可判定的。例如,不可能预先确定程序是否会停止运行。如果转换成生命游戏的术语,这就意味着这种构型是否会无限制的发展下去?的问题在形式上是不可判定的。于是就有了这样一个例子:我们知道万物理论,而且我们也知道一个可证明为根据此理论是无法回答的简单的问题。"

"完全正确",Nathan 表示同意,"那么你为什么认为现实世界的

真实的万物理论会是有意义的最终答案呢?"

"我不知道",Louise 长叹了一口气,她的信念一时发生了动摇。她摇了摇头,接着又兴奋起来。"这倒是有点虎头蛇尾的味道。"

Cohen-Kong 定理

容易查明,Langton 蚂蚁的万物理论是可时间反演的,也就是说,当前的图案及移动方向唯一地决定了过去与未来的情况。任何有界路径最终必定会重复同样的图案、位置及移动方向;根据时间反演性,这样一种路径必定是周期性的,无限地重复着同样的运动。因此蚂蚁到过的每个方格必定会被它无限多次地光顾。蚂蚁交替地作水平移动和垂直移动,因为它每在一格其方向都要变化 90 度。我们把蚂蚁水平地进入的方格称为 H 方格,而把它垂直地进入的方格称为 V 方格。H 方格和 V 方格就像棋盘的黑白方格一样铺满方格网。

选择蚂蚁到过的方格中处于最右上方位置的一个方格称为 M,也就是说,此方格之上的方格以及它右边的方格蚂蚁都没有去过。假定这是一个 H 方格。这样蚂蚁必定是从左边进入此方格,且向下离开此方格,因而 M 方格必定是白的。但现在 M 变成了黑色,这样当蚂蚁下次到达此方格时它将向上离开此方格,从而进入了一个它以前从未到过的方格。如果 M 是一个 V 方格,也会出现类似的问题。这一矛盾证明了不存在有界路径。

41. 细胞自动机的新家族：碎片、微滴、缺陷和精灵

有些数学系统虽然描述起来十分简单，但却能够产生出其复杂性令人难以置信的微型宇宙。这类微型宇宙不能用望远镜或空间飞船来探测，而只能用计算机来研究，所谓 Mandelbrot 数集便是其中一例。Mandelbrot 数集的元素是通过对简单的算术运算作反复迭代而求出的：但是，当这些元素在计算机显示屏上显示出来时，却构成了无比精巧的图像。单元自动机或许可以说是微型宇宙的更恰当的例子，因为这种自动机同我们现今所居住的宇宙一样是随时间演变的。

David Griffeath 发现了一类新的细胞自动机，它为这类微型宇宙提供了迄今为止最精彩的实例。这种新的细胞自动机从某一随机状态开始，终历四个不同的演变阶段，最后显示出一些奇异的晶体生长过程，它使人不能不联想到原始的生命形式。

细胞自动机由无限伸展的方格网构成，每一方格（即一个单元处于若干可能状态中的某一状态。随着一台想象的时钟每“滴答”响一声。所有单元的状态都按照一组简单的规则同时发生变化。在计算



机上执行这一过程时，各单元用显示屏上的像素表示，而不同的状态则用不同的颜色代表。只要有一组适当的规则和初始状态，以计算机为基础的细胞自动机就能产生出随时间而演变的不寻常的颜色图案来。

Griffeath 发明的单元自动机(下面我将把它称之为“循环空间”)依靠一条简单得令人发笑的规则，然而它却能产生出许多不但具有科学意义，而且非常漂亮的惊人现象来：这条规则的基础是将 n 个状态从 0 到 n－1 依次编号。如果某一单元在时钟“滴答”响一下时处于状态 K，那么它在时钟响下一声之前就必须“吃掉”相邻单元中其状态为 K－1 的所有单元。此时这种相邻单元的状态就由 K－1 变为 K，表示它已被状态为 K 的单元吞下去了。这种规则类似于自然界中的食物链：一个处于状态 2 的单元始终可以吞食一个处于状态 1 的单元，甚至当这个处于状态 1 的单元本身正在吞食 1 个处于状态 0 的单元时也是如此。但是，循环空间中的食物链是没有尽头的。因为处于状态 0 的单元可以吞食其状态为 n－1 的所有相邻单元(见彩图 16)。

以这样一条简单的规则为基础，循环空间自动机可以把随机的初始颜色分布转变为一些稳定的角形螺旋线。Griffeath 把初始的无序阶段称为“碎片”阶段，这样称呼是很有道理的：在这一阶段上循环空间自动机看起来显得乱七八糟，毫无头绪。然头，经过一段时间后，一些小小的微滴——微滴内部滚动着由各种颜色组成的阵阵波涛——取代了碎片。微滴不断长大，直至完全充满整个循环空间自动机。此时开始形成一些结晶状的螺旋线，这些精巧漂亮的螺旋线在旋转时不断长大。

每条螺旋线都是从 Griffeath 所谓的“缺陷”(这是从结晶学中借来的一个术语)中生长起来的。这些螺旋线开始时威风凛凛地慢慢长大，但最终它们之间将爆发争夺地盘的战斗。有些螺旋线的地盘



被他人所夺占,其他一些螺旋线则一直生存到在循环空间自动机的最后一阶段中占据统治地位。Griffeath 把这些生存斗争中的获胜者称为"精灵"。

单元的可能状态数无论是大是小、循环空间自动机的发展过程都是按相同方式展开的:首先是碎片阶段,然后依次是微滴阶段、缺陷阶段和精灵阶段。头三个阶段都具有亚稳态的特性,每一阶段都存在很长时间后(即存在许多个周期后)才让位于下一阶段。

试考虑一个其单元有 20 种可能状态的循环空间自动机。为什么随机的初始状态分布会是亚稳定的呢?换言之,为什么计算机屏幕上的大片区域需要很长时间才能改变其各种颜色随机混合而呈的灰色图景呢(诚然,开始时屏幕各处都有少数不引人注意的微滴在跳动,但它们似乎不起什么重要作用)?

这个问题的答案实际上非常简单。如果一个单元的状态恰好(比如说)是 5,那么它的相邻四个单元中至少有一个单元的状态是 4 的可能性有多大呢?很明显这一可能性是不大的。对于有 20 种可能状态的单元自动机,这一概率(既可能性)大约为 0.19。即使是它的相邻单元中有一个单元真的就是可供它美餐一顿的"食物",在时钟响了一声"滴答"时这一原始单元或被它吞食掉的那个单元的邻居中有一个单元处于状态 4 的概率又有多大呢?这一概率上的论证说明了在运行一个具有随机的初始状态分布的循环空间自动机时人们在计算机屏幕上所能看到的现象,这一自动机在零零星星地吃了几餐之后便停滞不动了。

然而实际上它并不是完全停滞不动。那几个孤零零的微滴现在的情况如何呢?一阵阵大鱼吃小鱼的浪潮在这些微滴的表面上席卷而过然后又沿新的方向从周围的碎片上反弹回来。微滴内的这类活动看来不仅仅是使它得以维持下去,而且实际上是使它不断长大:新的单元不断地从外围的碎片中进入微滴的阵营,参加了微滴内所发



生的活动。这些不断扩大地盘的微滴慢慢地融合在一起,从而使碎片的地盘最终只剩下几个零星的孤岛。究竟是什么原因使得微滴不断地长大呢?

微滴不断长大的原因之一是环圈的存在。环圈是细胞自动机内的一类特殊的闭合单元链,在这条链中每一单元的状态与其相邻单元的状态之差不大于1(见彩图17)。因此,如果在某一环圈中任意选定一个单元,那么只要在这个选定单元的状态上加上1或-1或0,便可得出下一单元的状态数(读者应记住,在循环空间自动机中最高状态与状态0之间只相差1)现在我们沿着环圈一个单元一个单元地走下去直到走完一周,同时把相邻两单元之间的状态数差不断地加起来,这样最后所得的结果很可能不为0。如果一条环圈具有这一特性,那么它就将使一个微滴无休止地不断生长下去。事实上这种环圈就是一个"缺陷",它正是使循环空间自动机的第三个亚稳定阶段得以出现的基础(见彩图18)。

作为缺陷的环圈所产生的单元活动与非缺陷环圈所产生的单元活动是大不相同的。缺陷使其周围的单元按照一种十分有规律的步调改变其状态。从计算机屏幕上看,围绕着缺陷的区域呈现出一种非常吸引人的、螺旋状的几何形态。最终,整个循环空间自动机(至少是我们在计算机屏幕上所看到的那一部分)将被这类螺旋形图案所完全占满。

当循环空间自动机从碎片阶段继续演化下去时,缺陷是如何出现呢? 自然,从这个单元式宇宙存在的最初瞬间开始,可能就已经有了一个缺陷了。事实上,如果把循环空间自动机看成是向四面八方无限伸展的一个宇宙,那么从统计分析上可以肯定它的碎片中存在缺陷。然而,通常的情况是缺陷是在先前并不存在缺陷的地方形成的。那么它是如何形成的呢? 我把这个任务留给读者,也就是要求读者作出一个由若干单元组成的小矩形,此矩形中不包含缺陷,但在



演变过程中它最终将产生出一个缺陷来。这就要求读者设计出某种简单的检验(只要不是让计算机运行得太久的),它能找出那些将产生出缺陷的区域。

第四个阶段——也就是最后一个阶段——单凭肉眼很难同第三个阶段区别开来。据 Griffeath 说,有些缺陷的"效率"比其他缺陷更高,这种高效表现在它们起着 n 态时钟的作用,也就是单元状态沿着最内环圈变化一周刚好需要时钟响 n 下。其变化节拍比这慢的环圈最终将被这些效率较高的环圈所吸收掉。这些效率较高的环圈就是盛行于循环空间自动机的第四阶段的"精灵"。精灵最终将无限期地存在下去,各个精灵与其相邻的精灵之间由一些看不见的,但却是坚实的栅栏所分开(见彩图 19)。

循环空间自动机是在单元自动计算机上运行的,它由装在一块电路板上的几个大芯片构成,此电路板又装在一台个人计算机中。在芯片上,单元自动机的每个单元由一个存储单元代表,存储内容的更新则由一个专用控制逻辑来实现。在全速运行时,单元自动计算机可以每秒 60 次的速度对 256×256 的单元阵列进行反复计算。

遗憾的是这台装置的费用比较昂贵。不过,读者也可以自己动手编写一种称为"DEMON"的循环空间自动机模拟程序(虽然它的速度比单元自动计算机要慢一些)。为此,读者可以用自己喜爱的程序设计语言把下面介绍的算法梗概改写为程序。这一算法使用两个数组(分别称为 new 和 old)来保持循环空间自动机各单元的当前状态与先前状态。算法梗概如下:

反复进行下列运算直至按键

对于 i←1 至 100

  对于 j←1 至 100

    对于单元(i,j)的每个相邻单元(k,i)

        如果 old(k,i)




```
=old(i,j)+1
  则 new(i,j)←
  old(k,i)
对于 i←1 至 100
  对于 j←1 至 100
    显示 new(i,j)
    old(i,j)←new
    (i,j)
```

上述算法中的外循环(当 DEMON 程序的用户按下某一键时此循环即中止)负责控制主要的检查与显示周期的反复进行。当然,也可以用其他方法构造这样一种循环。如果读者只有存储容量较小的计算机,那么最好是只使用较小的单元阵列(例如 50×50 的阵列)。上面算法中所用的两个内循环适合于 100×100 的单元阵列。单元(i,j)位于第 i 行和第 j 列的交点处。

DEMON 程序的最内层循环的作用是检查单元(i,j)的四个相邻单元中每一个的状态(因此读者在按上述算法改写出的程序中必须加入有关的赋值指令:当 i 等于 j 时,这些指令将 i-1 与 i+1 赋予下标 k,类似地,当 K 等于 i 时 j-1 与 j+1 必须赋予 1)。如果某一相邻单元的状态数恰好比单元(i,j)的状态数大 1,则单元(i,j)就被吃掉,也就是它的状态数变成了这个相邻单元的状态数。

DEMON 程序在计算 old(i,j)+1 的值时必须按模算术的规则进行计算。换言之,如果 old(i,j)恰好是 n-1,即达到了最高的状态数,那么 old(i,j)+1 的值就等于 0。因此,如果规定了单元空间(比如说)有 10 种状态,那么状态数将为 0,1,2……9,而 9+1=0。如果一个单元的状态数发生了变化,那么新得到的状态值就赋与数组 new(i,j)。

紧跟检查循环之后的双重循环显示出所有单元的保持在数组



new 中的状态值，然后对数组 old 作更新，即用数组 new 中的相应数值来代替数组 old 中的所有数值。

不但在计算单元状态数时需要应用模算术的计算规则，而且在计算下标 i 和 j 时也应当用这一规则。为了给人们造成一种单元空间是无穷无尽地伸展下去的印象，最简单的办法是使显示屏具有"返转"的特性。在具有返转特性的显示屏上，屏幕最右端的单元被认为是同最左端的单元相邻接的，而最下端的单元则被认为是同最上端的单元相邻接的。为了达到这样一种效果，下标值采用从 0 到 99 的一个数，而不用从 1 到 100 的 100 个数。紧挨着单元(23,99)的右边的那个单元实际上是(23,0)。因此，i－1，i＋1，j－1 以及 j＋1 这四个数——也就是单元(i,j)的相邻的四个单元的下标值——必须全都以模算术的形式表示。大多数程序设计语言都包含有自动实现此项功能的指令。

自然，DEMON 程序必须使它的用户能够在程序的控制下"预置"单元空间的初始状态。为了实现初始状态的预置，可以在算法中加入一项功能，以便按用户的要求规定状态数，此外还须用一个循环来给单元空间中的每个单元规定一个其值在允许范围内的、随机选定的初始状态数。

状态数究竟为多少才最合适呢？这完全取决于读者希望循环空间自动机的四个阶段之间需要多长时间才发生交替。如果状态数取得很大(比如说有 25 种以上的状态)，那么一个大小为 100×100 的随机选定初始状态的单元阵列很可能就永远固定下来、不发生任何变化了。另外，如果状态数取得很小，那么单元自动机各阶段之间替演替就会非常迅速，以致用户来不及细细品尝。Griffeath 建议将状态数选择在 12 至 16 之间。

42. 模拟进化：虫子如何捕食细菌

在一汪死水潭的肮脏的底部，一些原生动物在四处爬动，吃着像雨点般慢慢落下来的细菌。那些原生动物看上去全是一样的，但它们各自的行为大不相同。有些在觅食的过程中无目标地到处乱爬。结果没有吃到什么，而另一些却是目标比较明确，按照一种看上去似乎是井井有条的搜索方式在寻觅食物，因此它们便吃到许多细菌。这类微观世界有其独特的迷人之处，但上述原生动物乐园却更有一种特殊的意义：行为条理化的原生动物是在仅仅 1 个小时的时间内从行为无条理的同龄种群中进化而来的！

有些读者可能已经猜想到，这样一种微观世界不是在显微镜下观看，而是在计算机屏幕上观看的。产生这一图像的是一个叫做“模拟进化”的程序。极小的白色原生动物（Paimiter 把它们称为虫子）在屏幕上到处爬动，吞食着紫色的细菌。随着虫子一代一代的繁衍，我们可以观察到代与代之间觅食行为的进化。

牛津大学的 Richard Dawkins 也曾通过研究模拟生物进化过程的形态演变的程序来深入地认识生物进化。Dawkins 的程序所显示



的生物形态是由计算机产生的在有些时候类似于活的生物的图形。这些生物形态通过"人工选择"过程而进化。所谓"人工选择",就是计算机操作人员随机地从现有的生物形态的九种可能形式中选择出一种作为进化成未来几代生物形态的基础。

Dawkins的程序所产生的生物形态既稀奇古怪又十分有趣,有时甚至栩栩如生。但是,不能说它们是自然地进化而来的;也就是说,它们不是在一种内在的选择压力下进化的。然而,Dawkins认为有可能编写出一个计算机程序来模拟自然选择。计算机产生的这种有进化能力的物种将逐渐产生出越来越复杂的形态,而自然选择则可使形态数目减少到能够掌握的程度。此外,在这种选择中幸存下来的后代一定能够以其祖先完全不可能利用的新的方式进行进化(见彩图20)。

Palmiter的原生动物("虫子")肯定使我们向Dawkins所设想的目标更接近了一步。如彩图21所示,虫子(白色块)生活在一个矩形方框内,而细菌(紫色点)则不断地落在方框上。虫子要爬动并吃到细菌才能维持生命。每吃一个细菌,它就能够获得40单位的能量,此能量足够使虫子作40次移动。在细菌多的地方。一只虫子在几分钟之内就很容易获得1 500单位的能量。然而,如果出现了这种情况,另一种奇特的机制就会起作用:在这只虫子积蓄的能量下降到1 500单位以前,无论再吃多少细菌也不会使它的能量有所增加。

如果情况正相反,即一只虫子在很长时间内都找不到什么吃的,那么此虫子的能量储存就会逐渐降到0。这时,虫子便会无精打采地停止在原地,仿佛在为它的末日焦虑。几个周期后,它就像一盏小灯一样熄灭了。

当然,虫子能否成功地找到食物取决于它紧邻的细菌的多少。因为细菌比较均匀地沉淀在矩形范围内,所以,如果虫子的局部觅食已使某一处的细菌被吃光,那么别的地方的细菌一定会相当丰富。



一些虫子看来能比另一些虫子更迅速地到达细菌比较丰富的区域，这完全取决于虫子所作的移动——换句话说，取决于虫子的觅食方式。

对"模拟进化"程序的达尔文主义的解释，尽管抽象，但始终是围绕控制虫子移动方式的"基因"进行的。在真正的原生动物中，这些特殊的基因可能并不存在，但是 Palmiter 的程序所产生的虫子有六个这样的基因，分别用 F(前进)、R(向右)、HR(向极右)、RV(反向)、HL(向极左)、L(向左)标示(所有的方位都以虫子的观察点为基准。一般转弯 60°，极转弯是 120°)。虫子每作一次移动，其方向是按彩票摸奖那样的方式确定的，即从一种"数学帽子"(里面装着上述六个方向的签)中任意摸出一个签来。例如，如果程序摸到 L，虫子就向左转 60°的弯。摸到某一特定方向的概率是由赋予相应基因的一个数值确定的。因此，某个基因的值越高，该基因对虫子的总的移动方式所起的作用就越大。例如，如果虫子的 L 基因的值相对于其他五个基因的值较大，那么虫子大部分时间都是转向左边。

基因值的每种可能的组合形式都对应于一种移动方式，而且，一旦虫子的基因构成形式确定下来，便终生保持不变。用一种拟人化的说法就是，虫子只能希望它的后代胜过自己。

一只虫子作了 800 次移动后，便进入"成年期"，准备繁殖后代了。但是，只有处于"强壮"状态的虫子——也就是在其电子生物膜内已存贮了 1 000 单位以上的能量的虫子——才能进行繁殖。草履虫的繁殖过程是两个生殖细胞的接合，但程序中的虫子都是按分裂的方式繁殖，即一个强壮的成年虫子一分为二成两个新个体，每个新虫子的能量是其母体能量的一半。新虫子产生后，继承了母体的运动基因，但同时也出现细微的变异：每个后代的某一基因的值略有升高或降低。

例如，假设有一只强壮的成年虫子的基因值分别为：F = 3，R =



2,HR=0,RV=-2,HL=0,L=1。它的两个子代(分别表示为A和B)可能具有下列经过变异的基因组合：

A:F=4,R=2,HR=0,RV=-1,HL=0,L=1

B:F=3,R=2,HR=0,RV=-2,HL=-1,L=1

读者能够看出，子代A的F基因的值增大了1，而子代B的HL基因的值减少了1。

那么子代的移动方式与其母体有哪些不同呢？子代A前移的趋势比其母体稍强，而子代B作极左转弯的趋势比其母体稍有减弱。这些运动趋势的细微改变从计算机显示屏上观察，只有经过训练的人才能勉强看得出来。

最简单的“模拟进化”程序开始运行时，给10只虫子各赋予一种随机选定的基因组合，这就使得大多数虫子以一种无法预料的方式从矩形的一边到另一边紧张不安地跳动。一般说来这些跳“吉特巴舞”的虫子的死亡率很高。它们通常都是很快就吃光就近的食物，然后在这块没有食物的不毛之地上轻轻跳动，慢慢就饿死了。不过，也有些虫子得以幸存下来。

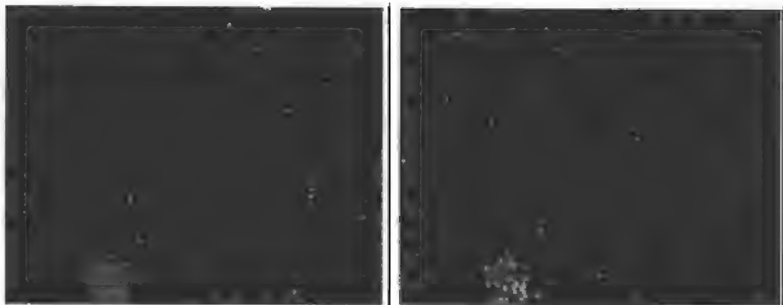


图　伊甸园(左)促进了“盘旋者”的进化(右)

时间一分钟一分钟地过去，虫子一代接一代地出现。这场小规模的生死斗争是引人入胜的，但是几分钟之后它的戏剧效果将真正

达到高潮。这时,观察者开始觉察到有些虫子的行为已发生了改变,它们不是颤抖,而是蹦跶。再过几分钟将出现打滚的虫子。20多分钟后,可以看到作滑行移动(至少是短距离的滑行)的虫子。这些虫子的运动方式看来比它们那些只会颤抖的祖先高明得多。的确,正是由于这个原因,它们当着人们的面堂而皇之地繁衍开来。

到一定时候,就会逐渐出现一些进行"巡航"的虫子。它们多数时间向前移动,但也不时转一下弯,这就意味着它们几乎总是朝前爬到细菌较为密集的地方。这种行为只要一在为数不多的几只虫子身上形成,很快就将在整个群体中占主宰地位,因为这些"巡洋舰"一样的虫子最终将吃掉绝大部分细菌。

尽管"巡洋舰"们构成了一类物种,然而在这个物种内部仍然存在一些变异。例如,有些"巡洋舰"右转弯的时候比左转弯的时候多。相反,另一些"巡洋舰"左转弯比右转弯多,当然还有偶尔作倒退移动的。有些"巡洋舰"大量繁殖适应能力很差的不良后代。一种最常见的遗传病是"盘旋病",患有这种病的虫子老是向一个方向转弯。这类不幸的虫子一般都是还没有尝到分裂繁殖的乐趣就一命呜呼了。

观察"巡洋舰"这个种群的出现已是十分有趣的了,但 Palmiter 的程序所展示的内容还不只这些。如果环境出现了变化,那么虫子将会如何进化呢?是否会有一个以上的物种进化出来?为了回答这个问题,我们可以在运行"模拟进化"程序时,让显示屏基本上保持不变,只是在其左下角的一块区域内放置特别多的细菌。这一区域内的细菌的补充速率远远高于正常的补充速率(见图),Palmiter 把这一细菌特别多的区域称为"伊甸园"。

虫子一代一代的产生,又一代一代的消失。"巡洋舰"种群的虫子如前所描述的那样不断地进化。但是,在伊甸园内却出现了完全不同的另一种情况。少数几只幸运的"跳吉特巴舞"的虫子无意中闯进了伊甸园。它们的无条理的寻食方式此时反倒帮了它们的大忙,



仗着这种寻食方式它们可以在伊甸园内游来晃去，不管晃到那里周围到处都是细菌，于是得以饱餐了一顿。

然而，随着伊甸园内的细菌越来越少，一种不利的环境压力开始起作用。靠晃动或跳吉特巴舞来捞取食物的虫子很快就发现它们的觅食方式不灵了，此时便正是"盘旋者"这类虫子大显身手的时机，盘旋病通常被看作是一种灾难性的遗传病，但在虫口过剩的伊甸园内它实际上却成了一个优点。的确，经过一段时间后，伊甸园里剩下的主要就是那些总是喜欢向着一个方向转弯的虫子了。其原因是很明显的：如果虫子在伊甸园内频繁地向同一个方向转弯（比如向左转弯），那么它存活的时间一般就比其跳吉特巴舞的祖先更长。

最多只要几个小时，伊甸园就几乎完全被高度特化的"盘旋者"所独占，我们不妨把它们叫做"神经质的轨道飞行器"，它们沿一个特定的轨道转上许多圈，然后突然向外跳动一下，又重新开始沿轨道运动，每移动一次就把路上的细菌捕食干净。

"模拟进化"程序是否能真正反映生物的进化模式呢？仅仅在非常有限的程度上可以这样说。这个程序揭示了环境是怎样有利于某些变异的存在与发展，从而最终导致新物种的形成。但是程序模拟的进化与真实的生物进化之间的类似也就到此为止。一旦一两种稳定的物种出现后，就不会再出现其他什么令人感兴趣的东西了。为了实现 Dawkins 关于无限的连续不断的计算机模拟进化的梦想，需要有什么样的模拟手段呢？也许至少需要一种计算机内的微型宇宙。

对那些有较高的编程水平、愿意自己来编写一个模拟进化的程序的读者，下面给他们介绍一下程序 BUGS(虫子)。

程序 BUGS 中的虫子由每边有三个像素的小正方形代表，图 1 示出了虫子能够移动的六个方向，下面这张简单的表规定了怎样根据虫子运动的方向改变虫子的中心像素的坐标。此表包含了两个有

六个元素的数组,即 x move 和 y move,

dir 0 1 2 3 4 5

x move 0 2 2 0 2 −2 −2

x move 2 1 −1 −2 −1 1

虫子运动的方向(即相对于计算机显示屏的运动方向)由变量 dir 的值给定。数组 x move 和 y move 中与 dir 相对应的一对数确定了虫子在一次移动时在水平方向和垂直方向上分别应移动多少个像素。例如,如果有只虫子指向方向 2,那么它必须向右移动两个像素,向下移动一个像素,因为 x move(2)=2,而 y move(2)= −1。

程序 BUGS 参照一个以每只虫子的遗传密码为基础的公式来确定它控制下的每只虫子的运动方向。虫子的遗传密码存贮在称为 gene 的一个二维数组中,数组元素 gene(k,j)为第 k 只虫子的第 j 个基因值。在上述公式里,每个基因的值都作为一个以 2 为底的幕的指数,以避免计算过程中出现负数。将以 2 为底、以各个基因值为指数的幂之和除 2 的 d 次方,即可求得虫子向方向 d 运动的概率。例如,虫子向极左方向运动的概率就是用 2F + 2R + 2HR + 2RV + 2HL + 2L 除 2HL。

程序 BUGS 按此方式计算出的概率值有六个,每个代表一种可能的运动方向。把这六个概率值加起来,其结果必然为 1。可以把这六个概率值看作是从 0 到 1 这个区间内的六个区段。换句话说,如果这六个方向的概率值用 P_0 到 P_5 来代表,则区段 0 即为 0 至 P_0 的区间,区段 1 即为 P_0 至 $P_0 + P_1$ 的区间,区段 2 为 $P_0 + P_1$ 至 $P_0 + P_1 + P_2$ 的区间,以此类推。

在每次循环中,程序 BUGS 都选择一个位于 0 与 1 之间的随机数,确定这个数落在哪一个区段内,并将区段序号作为变量 turn 的赋值,这样就为某一虫子的运动方向规定了新数。按此方法,对于 F,变量 turn 等于 0;对于 R,turn 等于 1;对于 HR,turn 等于 2;对于



RV,turn 等于 3,对于 HL,turn 为 4;对于 L,turn 为 5。

下面几个简单的语句是运动算法的结束语句:

dir←dir + turn(mod6)

bugx(k)←bugx(k) +

xmove(dir)

bugy(k)←bugy(k) +

ymove(dir)

数组 bugx(k)和 bugy(k)代表第 k 只虫子当前的坐标。

在第一句中,变量 dir 加上随机地确定的变量 turn 的值,使虫子的当前方向 dir 发生改变。加法必须是模加法;例如,如果 dir = 5(意即虫子向左上方爬),turn = 2(意即虫子要转极右弯),则新的 dir 值便为 5 + 2(mod6) = 1,虫子下一步将是向右上方移动。

程序 BUGS 必须按照这个公式来移动所有的虫子,每一步都要检查虫子是否遇上了障碍或者是否抓住了一个细菌。此外,程序 BUGS 必须保持对每只虫子的年龄和能量状态的记录,以便确定某个虫子是否应该被消灭掉或是否应该让它分裂繁殖。当一个虫子已有资格进行分裂繁殖时,程序就用两个新的虫子在原来位置上代替这个虫子。新虫子继承了老虫子的基因值,但是其中一个新虫子的某一随机选择的基因值增大了一定数值,而另一个新虫子的某一随机选择的基因值则减少了同一数值。

43."囚徒困境"与核战略

假定你现在是一位核大国的首脑,武装部队总参谋长递给你一张有点不好懂的数字表格。此表格中的两行标为"Us",而两列则标为"Them"。总参谋长向你禀报道:

"这个矩阵用数字形式汇集了我们的最杰出的军事、经济和政治思想的精华。您看,矩阵的格子内所表示的是双方可能采取的行动的各种组合;例如,如果我们发起攻击而对方按兵不动,我们的得分就是-63分……"

总参谋长迟疑了一会,可马上眼睛一亮道:

"然而敌方得分却是-74分。"

作为首脑,你习惯于发表讲演或剪彩之类的活动。当你出席了战略情况汇报会后,就不得不承认这样的会议颇为令人头痛。眼下,你正听着总参谋长继续往下讲:

"简言之,这是一个简单的2×2矩阵。这个矩阵给出了当其中一方发起攻击,双方都不发起攻击或双方都发起攻击这三种情况下双方得失的估计值。"

实际上,你现在已经进入了"After MAD"的起始阶段。"After



MAD"是一种核战略计算机对策,而 MAD 是"mutually assured destruction"(有把握的互相摧毁对方)的缩写。"After"指的是对策已发展到一个更高级的阶段:在这个阶段上,由于投入部署的核武器的命中精度越来越高,有把握互相摧毁对方的这种战略形势已经成为过去,先发动攻击的一方有可能把对方打得一败涂地,以致对方无法进行真正具有威胁性的反攻。结果,双方势必都企图采取先发制人的战略,不问青红皂白先下手再说。此时,双方都处于所谓的"囚徒困境"之中。下面将要详细介绍"After MAD"的内容。

前面已经指出,这个对策的关键是一个 2×2 矩阵。该矩阵之所以是 2×2 的,是因为对策双方都有两种选择,即要么进攻,要么不进攻。矩阵的内容随着对策的进行而逐步变化,因此每个矩阵的数字表示的是对策双方当前的战略力量对比。矩阵内的每一项为一对数字,每对数字与对策双方所采取的行动的某一可能组合相对应。前一个数字是该组合中"Us"的得分数,后一个数字是"Them"的得分数。例如,当"Them"攻击而"Us"按兵不动时,前一个数就比后一个数小。

双方根据当前矩阵的形势所采取的战略,当然就决定了下一个矩阵。每个矩阵可以引起另外四个矩阵的产生。到底引起哪一个产生,取决于对策双方采取的战略。在对策"After MAD"中有 110 个矩阵,也就是有 110 种可能的战略形势,但整个对策过程可以划分成五个不同的阶段:

第一阶段 MAD 阶段:双方战略武器仅对人口中心以及工业中心这类"软目标"有效。

第二阶段(MAD 阶段之后 10 年):多弹头分导再入大气层导弹(MIRV)部署在洲际导弹(ICBM)上。尽管大多数武器仍然对准软目标,但双方已经采用了所谓的打击军事实力战略:多弹头分导再入大气层导弹专门用于摧毁对方的洲际导弹发射井。

第三阶段(MAD之后20年):多弹头分导再入大气层导弹的准确性有了惊人的发展。由于现在抢先发动进攻的一方肯定能摧毁掉对方75%的高精度战略武器系统,所以MAD已经失去了它的意义。

		HE	
		合作	背叛
SHE	合作	(3,3)	(0,5)
	背叛	(5,0)	(1,1)

图1 “囚徒困境”

第四阶段(MAD之后30年):双方已经部署了可机动再入大气层导弹(MARV),这种导弹实际上能百分之百地摧毁掉对方的洲际导弹。这时,采用打击对方军事实力的第一次打击战略在军事上已具有巨大的吸引力,所以MAD已经完全没有意义了。对于这种进攻,威胁要对人口中心进行报复已不存在真正的威慑力。

第五阶段(MAD之后45年):空间弹道导弹防御系统(SBMD)能够摧毁对方的任何一次洲际导弹的攻击和多达50%的助推器。双方都有针对对方空间弹道导弹防御系统的空间雷达在靠近对方防御系统的轨道上运行。因此抢先发动进攻的国家能够摧毁对方的空间弹道导弹防御系统、人口中心以及洲际导弹,而自己的空间弹道导弹防御系统却能不受一点损伤地保存下来,从而可以阻挡敌方随后用潜射导弹发起的反攻。

这五个阶段的相对不稳定性反映在五个核心矩阵上(见图2)。只要双方都不发动攻击,就可以从一个阶段的核心矩阵变到下一阶

段的核心矩阵上。然而,随着战略力量发展到相当的水平,抢先发动进攻的好处就日益明显,并且双方都感到对方也可能想进攻自己,所以双方发动进攻的欲望都在与日俱增。最后的两个矩阵已经属于"囚徒困境"的情况了。

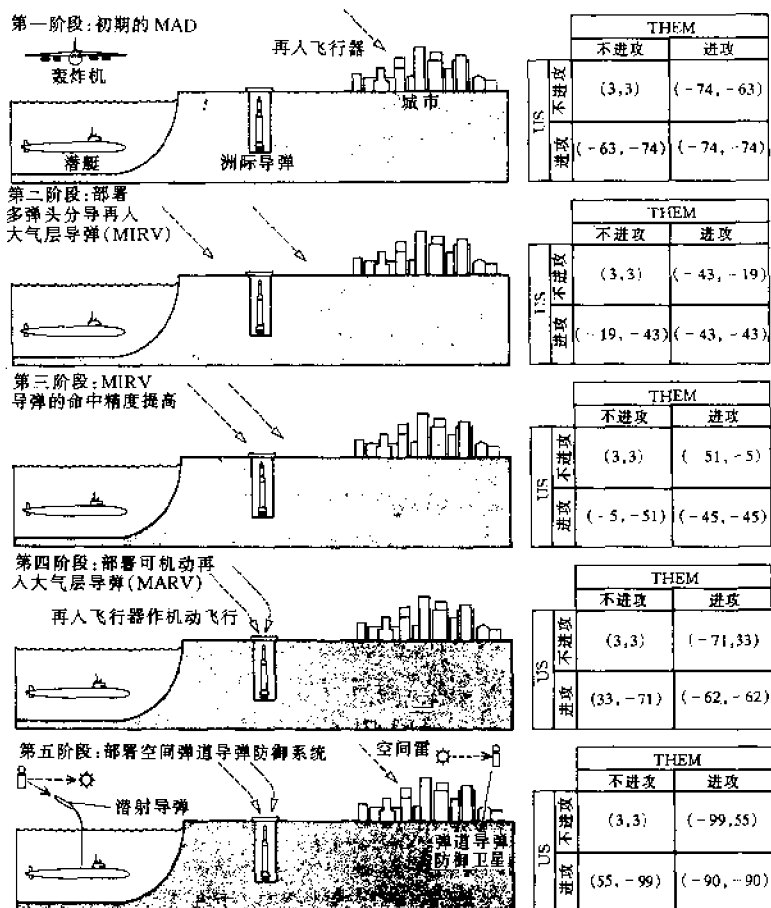


图 2 "After MAD"的五个阶段和代表这五个阶段的核心矩阵



在“囚徒困境”中，两个囚徒可以“合作”，即都否认与案件有关，也可以互相“背叛”，即供认事实真相。图 1 是关于“囚徒困境”的标准矩阵。把这两个囚徒称为“He”(他)和“She”(她)。如果 He 和 She 合作，那么他俩得到 3 分，如果两个都背叛了对方，他们就各得 -1 分，但如果只是一方背叛，那位背叛者就能得到 5 分，而被出卖者得 0 分。

那么在“囚徒困境”对策中什么是最好的策略呢？要估计出“囚徒困境”的最佳策略，仅靠一局交锋是不行的。得通过许多局的交锋才能估计出来。在对策进行了若干局后，所得收益的平均值就反映出了某个策略的好坏。

在一次“囚徒困境”邀请赛中(参赛者是体现了各种不同策略的计算机程序)，每对程序作 1 000 次比赛。获胜者是一个叫“针锋相对”(Tit for Tat)的非常简单的程序，这个程序采取的策略是对方程序怎么行动，自己就怎么行动，仅此而已。

“After MAD”中的矩阵与“囚徒困境”中的那些矩阵相似，只是用“hold(按兵不动)”和“attack(进攻)”分别代替“cooperate(合作)”和“defect(背叛)”。此外，“After MAD”矩阵中的数字通常更大，比如第四阶段的核心矩阵上的数字(见图 3)。这个矩阵中的各个数字间存在一种关系(所有的“囚徒困境”对策都有这一特点)即：“抢先进攻”的得分(33 分)，“和平相处”的得分(3 分)，“同时进攻”的得分(-62 分)与“坐等挨打”的得分(-71 分)，这四个得分构成一个严格递降的序列。人们可以期望在“After MAD”中对抗的双方都采用针锋相对的策略，从而保证双方不仅有一个合情合理的得分，而且(象征性地讲)也保证了世界和平。

应该说明的是，参加这个对策比赛的学生事先都得到过详细的指点，并且要求他们写出每步行动的理由。但是，既然世界本身可以看作是对实际事物的一场巨大试验，也许我们已经算是很幸运的了。



下面介绍此对策的矩阵是如何建立的，以及比赛者对这些矩阵所表现出的战略形势作何反映。

		THEM	
		不进攻	进攻
US	不进攻	(3,3)	(−71,33)
	进攻	(33,−71)	(−62,−62)

图3 "After MAD"第四阶段的核心矩阵

在"After MAD"的每个矩阵中，每个得分由三部分组成：军事力量、经济与社会利益以及政权。每部分的基本单位为1分。三个基本单位之和为3，这是最小的正收益，它表示一个没有战争的国家在对策的每一步所代表的一个短时期内将在这三个方面有所发展，但发展的速度不快。这三个部分对总得分所能作出的最大贡献(无论正或负)都是33个单位。

下面我们来看怎样确定第三阶段的核心矩阵中"抢先进攻"(即一方进攻而对方没有还击)的得分的。

军事力量这一部分最初的估计是+5分，但是考虑到对方如果还击，自己的军事实力就会被削弱一部分。所以，这5分中得减去3分。3是这样得出的：如果一方发起全面攻击，就可能摧毁对方75%的战略武器。这样先发起进攻的一方就使对方损失大约24分(75/100×33)。遭到攻击的一方仅剩下了25%的武器。因此，它用剩下的这点武器发起反击只能使挑起战端者损失6分(25/100×24)。考虑到用这25%的武器进行反击并不是完全有把握的，所以这6分还要打一半的折扣，于是就得出了3分。

从5分中减去3分(即对方报复时自己所付出的代价)就剩下



+2分,然后把这2分加在另外两个部分的得分上。假定经济及社会利益的得分为-2,政权的得分为-5。发动战争的国家由于对方的反击可能要损失掉一部分工业及人口;此外,在一个主张正义的世界上,发动战争的行为也要遭到世界舆论的一致谴责。所以,先发动进攻的一方的得分总计是+2-2-5=-5分。

在"After MAD"比赛中由计算机打印出来的说明可以使比赛者清楚地了解目前形势的战略意义。由于有110个矩阵,因此也就有110个说明。当对策进行到某一矩阵时,相应的说明就用文字显示给对方。

只要比赛的一方(或双方都)背信弃义向对方发起进攻,战争就不可避免了。此时,比赛就从战争爆发时的核心矩阵发展到某一个分支矩阵上。战争可能是以双方同时毁灭而告终(这样比赛也就结束),也可能是使双方都退却到先前的某个矩阵上以收拾残余重振旗鼓。

尽管双方都知道他们应当尽其所能来增加自己的得分,但看来他们仍然考虑了其他许多因素。由于他们不能互相见面或互相联系,以致他们有时候会变得有点想入非非。下面我们来看看简称为Column和Row的两位比赛者是如何经受住"After MAD"的考验的。

Row的表现完全是一个和平主义者。正如他的计算机记录所显示出的那样,他并没有竭力增加他自己的得分,而是试图按道义原则行事。"我参加比赛并不是为了分数。我准备在多数情况下都采取信任的态度来对待对方,我将始终尽可能寻求合作……我希望对方也采取我这样的态度"。

Column开头几步的记录表明他没有采取背叛行动,因为那样做并不能给他带来多少好处。他除了想尽量增加他的得分之外,就想尽量拉开与Row的比分的差距。从一开始他就显得有些担心Row会向他进攻。过了一段时间,Column终于迫不及待地向Row发动



进攻。这下 Row 感到很为难了。他想,Column 之所以进攻他,有两方面的原因。一是他怕 Row 先进攻他,二是他想使比赛更有趣些。于是 Row 希望 Column 会因害怕遭到报复而不再发动攻击。

在接下来的几步中双方再度合作,这样 Row 就更加相信 Column 不会再向他进攻了,因为他已经用行动证明了自己的诚意。但是 Column 对比赛情况的理解与 Row 完全不同。实际上,Column 断定 Row 正在暗中恢复他的力量,一旦 Row 能够向自己发动有效的进攻,他就会下手了。因此,在下一步时 Column 再度向 Row 发动了攻击。他对 Row 的不断合作甚至变得有点神经质了。他想:"最终我要把他逼到即使他对我进行还击他也只有化为乌有的地步"。这一轮比赛结束后,Row 反省了自己的策略,确信有必要作些修改。他的新战略仍然遵守道义准则,但却明显具有了"针锋相对"的特点。"我的新战略的基本方针是'不首先使用核武器(NFU)'。如果对方有背叛行动,我将立即进行报复直到他重新开始合作或者清楚地表明他将不惜一切代价去取胜时为止。在这两种情况下我都会主动再度合作,因为如果他决心屠杀我的人民,我也没有必要把他的人民都杀光。只要对方出现背叛行为,我就采取 NFU 战略,直到进入了 MARV(可机动的再入大气层导弹)阶段为止。一到了这个阶段,我就要采取背叛战略,以免遭到单方面的攻击而失去防御能力……尽管这一策略显得有些不近情理,但看来却是确保人类生命财产少受损失的最好途径"。

根据学生们在比赛中表现的行为,可以看出,比赛者并不是按他们所得到的指点那样理解矩阵的收益,而是按别的标准去理解这一收益的。有时,比赛者竭力想把他与别人的比分的差距拉大,这就说明了为什么许多人一开始就采取背叛行为,而有时比赛者是想尽量增加自己的得分,这说明了为什么有 20% 的人从不背叛。有的人追求战略优势地位,有的人则与之相反,纯粹按道义准则行事。



一位学生在一篇文章中概括地讲了他对这个对策的感想，这里我摘引他的最后几句话："这个对策模拟出的那种'诱惑力'之大，足以使一些学生互相对对方发起大规模的进攻。这对于力量均势(恐怖均势)理论来说并不是一个好兆头。如果世界领袖们开始在视频终端上观看他们的竞赛场，那我们可能全都卷入了一场 PACMAN 游戏中，所有的点子都被吃掉，而兑换角子的机器中将一个钱都不剩下。"

44."忙河狸"最勤奋的图灵机



在现存的动物中，除了蜜蜂以外，河狸也许是最忙碌的动物了。它们整天都在宁静的北部水域中忙个不停，把树枝拣回到自己的堤坝那儿去。毫无疑问，正是这种行为使得俄亥俄州立大学的 Rado 把一种图灵机问题命名为"忙河狸"游戏。早在 60 年代初，Rado 就想过这个问题：一台图灵机到停机时究竟能够打印出多少"1"来。特别是一台有几种可能状态的图灵机，如果对一条全是"0"的纸带开始操作，那么在停机之前最多可以打出多少个 1 来？当 $n=1$，$n=2$，$n=3$，$n=4$ 时答案是肯定的，但是当 $n=5$ 或者大于 5 时，答案是未知的。

1983 年在前联邦德国多特蒙德举行了一场竞赛，看看谁能发现具有 5 种状态的最忙碌的河狸。在竞赛举行的前一年就写好了作为候选者图灵机的程序，并且发展了相应的硬件来检验这些机器。在这项工作的进程中，发现了一些行为古怪的河狸，因而使 Castor（河狸）发展成为一种具有动物学家迄今还不知道的多样物种的属。

一台图灵机包括一条无限长的纸带，一个在纸带上读出和写入

符号的读写头，一个具有有限个内部状态的控制装置（见图1）。这些部件可以看成是该装置的硬件，而控制装置的内容则是软件——图灵机程序。正是该程序使得一台图灵机有别于其他的图灵机。这种程序是一张表格，机器就根据它来确定下一步要采取的行动。对于控制装置的每一种可能状态和纸带读写头目前位置上可能出现的每一种符号，在表上都有一项相应的指令，告诉机器在纸带上打印什么符号，读写头向什么方向运动以及下面要进入的状态。本文所讨论的所有图灵机都是从状态1开始。

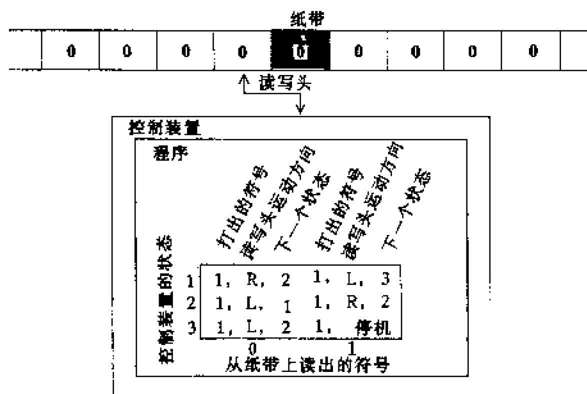


图1　一台图灵机及其程序

通过把控制装置的状态以及纸带（或纸带上的某一区域）上的符号连续地记录下来，可以跟踪图灵机的活动；还应当指出目前扫描到记录带上的哪一格。图2就是跟踪图1所示的图灵机。图2中的每一行都是对该机的一个“瞬时描述”。规定纸带的两头是无限长的，并且规定当机器转移到最终状态（即进入停机状态）时允许打印一个符号。本文所采用的瞬时描述的格式是与忙河狸竞赛中所采用的格式相一致的。

状态	纸带								
1	0	0	0	0	0	**0**	0	0	0
2	0	0	0	0	1	0	**0**	0	0
1	0	0	0	0	1	**1**	0	0	0
3	0	0	0	0	**1**	1	0	0	0
2	0	0	0	**1**	1	1	0	0	0
1	0	0	**1**	1	1	1	0	0	0
2	0	1	1	**1**	1	1	0	0	0
2	0	1	1	1	**1**	1	0	0	0
2	0	1	1	1	1	**1**	0	0	0
2	0	1	1	1	1	1	**0**	0	0
2	0	1	1	1	1	1	0	**0**	0
1	0	1	1	1	1	1	**1**	0	0
3	0	1	1	1	1	**1**	1	0	0
停机	0	1	1	1	1	1	1	0	0

图 2 跟踪图灵机操作的"瞬时描述"

一个具有 n 种状态的忙河狸要成为一台 n 态图灵机必须满足两个条件。第一,如果它是从一条全部为 0 的纸带开始操作,它最终必然会停下来;第二,它写下的 1 至少和会停机的其他 n 态机一样多。图 3 显示了有一种状态和三种状态的忙河狸。每一台图灵机都用一张状态转移表来表示,其中每个状态是一个标号的圆圈,而状态之间的转移则用箭头来表示。箭头上所标的字母和数字描述了图灵机的动作。例如,假设三种状态的忙河狸处于状态 1,读写头的纸带上读

出 0。在这种情况下图灵机就按标有"0,1,R"的箭头进行转移,转向状态 2。因而机器在读出 0 之后,在纸带上写下 1,读写头向右移动一格并进入状态 2。

一台 n 态图灵机停机前能够产生的 1 的最大数目用 Σ(n)表示。如前所述,只有对于 n 的最初 4 个值,Σ(n)的值才是已知的。单态忙河狸在停机前只写一个 1,换言之 Σ(1)等于 1。二态忙河狸产生 4 个 1 的序列。读者能否设计出这样的机器来? 三态忙河狸写出 6 个 1;图 2 显示了一台三态河狸机的程序和瞬时描述序列。图 3 显示了它的状态转移表。三态河狸机是由贝尔实验室的 Rado 和 Shen Lin 在 1962 年发现的。1973 年,波恩大学的 Bruno Weimann 发现了一种四态忙河狸,它的输出是 13 个连续的 1。从那以后,理论家们就一直在寻找五态忙河狸。

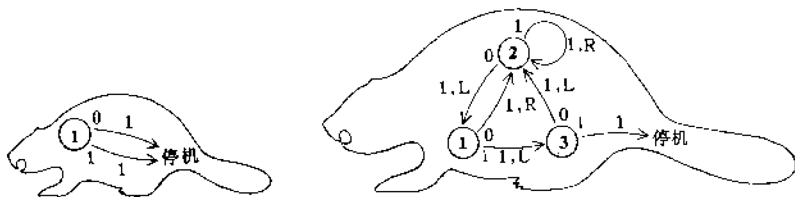


图 3　单态与三态忙河狸

1983 年 1 月在一次理论计算机科学大会期间,在多特蒙德大学举行了一次忙河狸竞赛。约 133 台五态图灵机参加了竞赛。汉堡的 Uwe Schult 获胜,他的机器在停机前产生了 501 个 1。图 4 显示了获胜计算机的状态转移表。亚军是巴登的 Brown Boveri 研究中心的 Jochen Ludewig,他的图灵机打印出 240 个 1。

Schult 的图灵机是忙河狸吗? Schult 和 Wankmuller、Ludewig 一样,猜想它是忙河狸。换言之,他猜测没有一台五态图灵机在停机之前可以产生多于 501 个的 1。这种猜想如何才能得到证明呢? 答



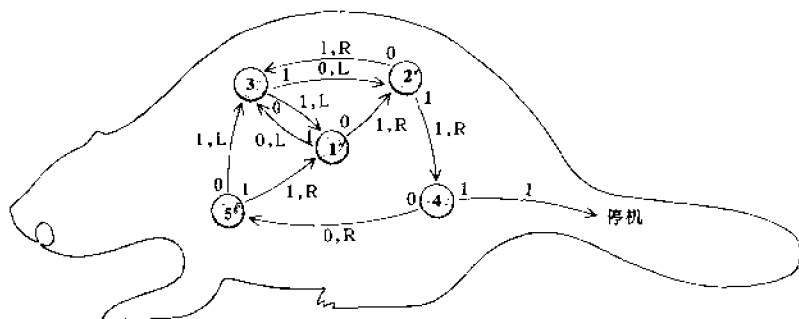


图 4　Uwe Schult 提出的五态忙河狸的候选者

案的依据是用计算机进行彻底的搜索,Schult 采用这种搜索方法首先发现了使他获得锦标的图灵机。在叙述 Schult 如何用自己的计算机捕捉五态忙河狸之前,我们先更加详细地考察一下函数 Σ(n),以便了解为什么忙河狸游戏即使借助于计算机来玩也是如此困难。

函数 Σ(n)有一种不寻常的性质:它是不可计算的。它的增长速度实在太快了。从它的最初 4 个值,即 1,4,6,13 来看,增长的速度还是一般的。如果五态机能打印出的 1 的个数的最大值的确是 501,那么 Σ(n)的增长看来似乎并没有超过指数函数。Schult 已经发现了一种六态图灵机能产生 2 075 个 1,这再次表明增长的速度还是容易处理的。然而,Schult 也发现了一台十二态机能产生如此多的 1,以致这个数字要用下面这个令人头昏脑胀的式子来表示:

4

4 096

·

·

·

4 096

4 096



6×4 096

在这个式子里,4 096 出现了 166 次,式子中的省略号代表 162 个 4 096。这个式子可以从上往下计算:先算 4 096 的 4 次方,然后根据计算的结果求 4 096 的如此多次乘幂,再根据计算的结果求 4 096 的乘幂,依次类推。算到最底下一行后再乘以 6。

如果有谁对于这样长的一串 1 不感到头疼,那么欢迎他再构造一个更大的数目。你可以用乘法或乘幂的方式写下自己所喜欢的任何一个式子;你也可以用 n 来代替其中的任何数字。不管你写出什么样的式子,当 n 足够大时,n 态忙河狸所产生的 1 的数目总可以大于根据该式子计算出来的值。由此可见,对于任意大的 n 值,Σ(n) 是不可计算的。最好是只对 n 的某些确定的较小值来计算 Σ(n)。

玩忙河狸游戏常常借助于计算机是不足为奇的。其基本方法是系统地考察所有的 n 态图灵机。每造出一台机器,就模拟它在开始时全为 0 的纸带上的行为。如果计算机停机时运算的步骤没有超过指定的数目,那么就把打印出的 1 的数目与迄今为止所发现的"最忙的"图灵机记录进行比较。这样就可以逐步发现新的优胜者。

用这种方法搜索 n 态忙河狸有两个重大的缺陷。首先,可以构造的图灵机的数量是巨大的,例如五态机就有 63 403 380 965 376 种。其次,我们不知道到停机为止要等多长时间;n 态机的状态转移的最大数目(仍然是一定要停止的)可以用函数 s(n)来表示,它本身就是不可计算的数字。显然,s(n)的增长甚至比 Σ(n)更快,因为图灵机每打印出一个 1 就必须作一次状态转移。计算 s(n)就相当于解决图灵机的停机问题,这是图灵证明为不可判定的第一批问题之一。

Schult 在 1982 年把苹果Ⅱ型个人计算机用来捕捉忙河狸。他在原来的中央处理机上增加了一块电路板,上面装有 Motorola 微处理机;他用这台辅助处理机的机器语言写下了自己的搜索程序。为



了检验由这程序产生的大量图灵机，他把标准的电子元件安装在另外一块电路板上，然后插进苹果Ⅱ型机中去，这样就作出了一台实际的硬件图灵机。这个装置提供了一条有4 096个方格的模拟纸带，还提供了一些用于存贮程序以及图灵机的当前状态和打印头位置的寄存器。据Schult估计，如果没有这样的专用硬件，他的搜索工作就需要20个月的机时。即使用这种硬件来扩充苹果Ⅱ型机，也仍然用了803小时的机时才找到获胜的图灵机。

在设计必要的软件时，Schult还因为使搜索程序与图灵机的硬件密切相关而得益。该程序以所有可能的方式系统地填满五态图灵机的转移表。在一张表填完之前，就把它拿到图灵机硬件上去进行检验。在许多情况下，一张未填完的表所规定的图灵机，在达到任何未规定的项之前就已经用完了运算时间或存贮空间。因此，必须抛弃这张未填完的表以及把它填完后的所有可能形式。

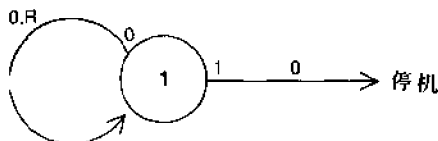


图5　纸带清洗机

虽然Schult在很大程度上解决了如何处理大量图灵机的问题，但是可以说，他对于五态忙河狸的停机问题的处理是不够严密的。由于缺乏关于s(5)——五态图灵机在停机前可能进行的状态转移的最大值——的确切信息，这个数字只能靠猜测。Schult把可能进行的状态转移数限定在500 000之内；也就是说，他采用的假设是，如果一架机器在作了500 000次状态转移之后仍然没有停机，那它就永远不会停机了。他还必须对候选的忙河狸在空间上也加以限制；因为模拟纸带只有4 096格，而且他的图灵机总是从这条有限长



纸带的中心开始，所以，如果一个候选者离开初始位置越过了 2 048 格，那就被看成是一个“犯规者”。一架犯规的图灵机不但不会停机，而且不断地、无限制地达到纸带上的新格子。

有 133 台图灵机参加了多特蒙德的竞赛，只有 4 台机器产生的 1 多于 100 个。每台图灵机的操作都用 Siemens 7.748 型计算机来模拟。要确定优胜者需要 1 个多小时的处理机机时。

亚军 Ludewig 用 Pascal 程序语言写了他的搜索忙河狸的程序，并在数字设备公司制造的 VAX 型大型微处理机上进行运行。尽管对于候选的图灵机作了更精巧的分析，为了发现他的入选者仍然花费了 1 647 小时的中心处理机机时，入选的图灵机产生了 240 个 1。毫不奇怪，Schult 也发现了 Ludeewig 的机器；同样有趣的是，他还发现在 Ludewig 的机器和他自己的机器之间再没有别的机器。显然，任何会停机的五态图灵机打出的 1 如果超过 240 个，那就至少是 501 个。

Ludwig 在自己搜索的过程中发现了一些奇怪的图灵机，它们的行为与河狸很相似。除了打印 1 以外，还有其他的方式可以使河狸去忙个不停。例如一台图灵机可以不去打印 1，而是从开始的那一格起移动相当一段距离，然后再停机。也可以既不打印 1，也不移动相当长的距离，而只是进行许多次状态转移然后再停机。在参加多特蒙德竞赛的机器中，Schult 的机器在这三方面都取胜了。

在这些河狸中有一部分是三态的，看一看这些三态河狸是很有趣的。要找出它们的任何尝试都会从使用计算机（个人计算机或其他型号）而受益，即使只是检验一个人头脑中设计的图灵机程序也是如此。

图灵机的模拟程序并不难写。可以用一维阵列来表示纸带；该阵列的内容只有 0 或 1，它们可以在计算机的屏幕上显示出来。如果再指出读写头的位置，那么这种显示方法就可以提供大量的信息。



例如,正在被扫描的符号的正下方就可以直接显示出机器目前所处的状态。

要显示图灵机的程序就要采用2维阵列。该阵列的每一个元素都是一组机器指令。对于控制装置的每一种状态和每一种可能的纸带符号,都必须规定机器指令。对于三态图灵机来讲,这种阵列为3行2列;它的结构恰好就是图1所示那种程序的结构。机器的每种状态规定了1行,而读写头下的纸带符号规定了1列;在指定的行和列的交叉点的指令规定了图灵机下一步的行动。

假设机器的状态是1而纸带上的符号是0。于是模拟程序就查阅阵列的第1行和第0列,结果发现指令是"1,R,2"。于是机器就在纸带上写下1,把读写头往右移,一格并进入状态2。执行这种指令的一种方式就是规定3个变量:STATE(状态)、HEAD(读写头)和SYMBOL(符号)。在每一周期的开始,STATE与SYMBOL的值决定了机器到表中的什么位置去寻找下一条指令。所找到的指令的第一部分(在本例中是1)写在纸带上;第二部分(R)就成为HEAD的新值,而第三部分(2)就是STATE的值。然后读写头(根据HEAD值所指示的方向)移动,而在新位置上发现的值就作为SYMBOL的值。于是再次开始这样的循环。

可以采取名称更加方便更加有效的策略来编制这种方案的程序。例如L和R可以用数字来代替,因为一般来讲,数字更便于计算机操作。此外,在程序中对于导致停机状态的转移也要作特殊的处理。

可以用图灵机的模拟程序来检验你对下述小难题的答案,但这决不意味着要解决这种难题非得用这程序不可。

设想你从当地的计算机商店买了一些用过的图灵机纸带。在把你的忙河狸放到这些纸带上之前,必须先把这些纸带洗干净;因此必须把上面所有的1都变成0。你决定设计一架简单的图灵机来代替

你做这项清洗工作。

这些磁带中有一条上面只有一个1,而其他的格子都是0。你必须创造一架图灵机去找到这个1,把它清洗掉(把1变成0),然后停机。当然,如果你的洗带机具有的状态越少,它就越漂亮。图6所示的那种纸带清洗机就是非常漂亮的。不幸的是它只有一半时间是在工作!

其余的纸带也和第一根一样,只是在它们上面有更多的1,但已经知道,在每一种情况下1的数目都是有限的。你能否构造一架纸带清洗机,它能把所有的1都变成0?当然它是永远不会停机的。

45. 图灵地铁的故事

我和 Tweedle 孪生兄妹拉着吊带站在纽约市的地铁车厢里。人们一般称 Delia Tweedle 为 Dee,而她的兄弟自然也就成了 Dum,虽然他的真正名字叫 Seymour。像往常一样,他们互相打断对方的讲话。

"嗯,如果宇宙是算法性的,那么高度的人工智能——"

"别卖弄你的学问了,Dee,你指的是会思维的计算机——"

"从原则上说肯定是可行的。"

"为什么?"我说。

"如果我们的宇宙是算法性的——"

"你就可以建造一台计算机来模拟它——"

"因而此计算机将模拟宇宙中的所有东西,包括正在进行这场谈话的我们",Dee 得出结论道。

"你是否认为如果你是正确的,那么一套足够复杂的地铁系统就可以具有智能?"我说。"它将思考得相当慢,但它仍然能够思考。"

"这太愚蠢了",Dee 大叫起来。"地铁是不会思考的。"

"或许不会。不过,我刚刚读到一篇非常迷人的文章,据这篇文



章说，地铁能够进行计算。此文讲的是关于列车系统的计算能力。”

“你说的是玩具列车系统吗？就是那些轨道、岔道口以及两侧画有绵羊的隧道？”“不错，Dee。凡是列车系统能做的事，地铁肯定也能做。它并不完全是你那种超并行的超超超级计算机，但是在理论计算能力方面，它可以说是毫不逊色。毕竟计算机仅仅是一个由适应性开关构成的巨大的开关电路，而列车则可以借助岔道口来转换轨道。问题是：如果你有足够的轨道（直轨和弯轨）、桥梁及各种各样的岔道口，但却只有一台机车且没有其他铁道车辆，那么当你设置适当的轨道布局后能够进行些什么样的计算呢？”

“我看不出列车怎么能进行计算”，Dee 困惑地说，“它不过是装在轮子上沿着轨道奔驰的东西罢了。”

“电子也不过是沿着电线运动的东西，然而计算机正是借助电子来进行计算的，”我指出，“在这两种情况下，计算工作都是一个如何解释的问题。对于轨道系统来说，就是要把输入编码为 0 和 1，它们对应于各种岔道口的扳道位置。然后就可以开动列车穿过这一系统。有时这些扳道位置会发生变化，此变化又会改变列车的运行路径。最终列车会进入一条通向终点站的侧线，于是程序就‘停止’，而你就可以从这同一批岔道口的扳道位置上读出输出了。”

“不错，我觉得这似乎是可行的。但它的确可行吗？”

“让我从最简单的转辙装置——称为‘懒惰岔道口’——入手。这是一段 Y 字形的轨道。列车从下方进入 Y 字形轨道，沿竖直的一段轨道上行，然后从岔道口扳到的那一段分支驶出，而整个 Y 字形轨道不发生任何变化。但是，从两条分支中的一条驶入的列车，如有必要，将会扳动道岔使该分支与竖直轨道相连，然后列车从竖直轨道驶出（见图 1）。懒惰岔道口有两种状态，取决于哪条分支与竖直轨道相连：我们称这两种状态为‘左’和‘右’。”



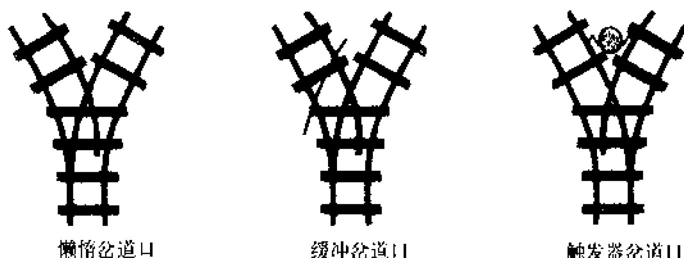


图 1　三种类型的岔道口

"接下来的一类岔道口是缓冲岔道口。它与懒惰岔道口相似,其不同之处在于沿 Y 形轨道的竖直段进入的列车总是经同一分支轨道驶出。第三类岔道口是触发器。"

"对于触发器岔道口,列车总是沿 Y 形轨道的竖直段进入,并轮流穿过左侧和右侧分支轨道驶出",我继续说道。我们在列车的震颤声中摇摇晃晃地进入了自由大道车站。"主要的问题是,有了这些部件,我们能建造一台计算机吗?"

"什么类型的计算机?"Dee 问题。

"图灵机",我说。"图灵证明"了他那简单的计算系统模型能够做可编程数字计算机所能做的任何事情。设想图灵机是一个可以沿着非常长的方格纸带移动的框,每个纸带中有一个符号(0 或 1)。你可以使用无限长的纸带;如果你不喜欢无限的话,你只须随时准备好在需要更多的方格时将这些方格加到纸带中去。

"这个框可以取有限多的一组内部状态中的任何一种状态,取决于框内是什么硬件。对于框本身的状态以及框下方的方格内打印出的数字的每种组合方式,这个框必须执行一小组指令,如:

'让纸带上现有的数字保持不变/改变这个数字。'

'然后向左/右移一格。'

'然后进入某一规定的内部状态,准备好开始进行下一步操作。'

或者该指令可以就是一个‘停止’，于是计算就终结了。”

“给我举个例子看看。”

“好的。对于有 3 个状态——分别称为 1,2 和 3——的图灵机，其典型的规则是：”

‘状态 1,数字 0’：改变数字，向左移一格，进入状态 2。

‘状态 1,数字 1：停止。’

‘状态 2,数字 0：保持数字不变，向右移一格，进入状态 3。’

‘状态 2,数字 1：改变数字，向左移 1 格，进入状态 2。’

‘状态 3,数字 0：改变数字，向右移 1 格，进入状态 1。’

‘状态 3,数字 1：保持数字不变，向左移一格，进入状态。’

这能计算什么呢？”

“对此我连最模糊的概念都没有，Dum。试试看吧。毕竟，实用的程序所需的状态数一般都远远超过 3 个。纸带上的数字是计算机的输入。对框的哪些内部状态执行哪些指令的指令表构成了程序，而计算终止时纸带上的数字构成输出。令人感兴趣的是，这些简单装置能够执行任何算法。所以我们需要的就是找出一个能模拟任何选定的图灵机的列车系统。”

“这事相当棘手。”

“是的。把这个问题分解成若干阶段来解决比较有利。现在我们的想法是要找出一个能够起到框的作用的列车系统。我们不用纸带，而是把许许多多的这种框并排起来彼此接通，以此代表整个纸带(见图 2)。每个框有若干条轨道从左侧进入，又有同样数目的轨道从右侧出来——每个内部状态由一条轨道代表。”

“这样就不是框沿着纸带移动——”Dee 开始说道。

“而是列车沿着这一行框运行”，Dum 热情地把话接着说完。

“你可以说出‘纸带’上的哪个‘方格’正在被——”

“列车所在的哪个框进行处理。真是干净利落。”

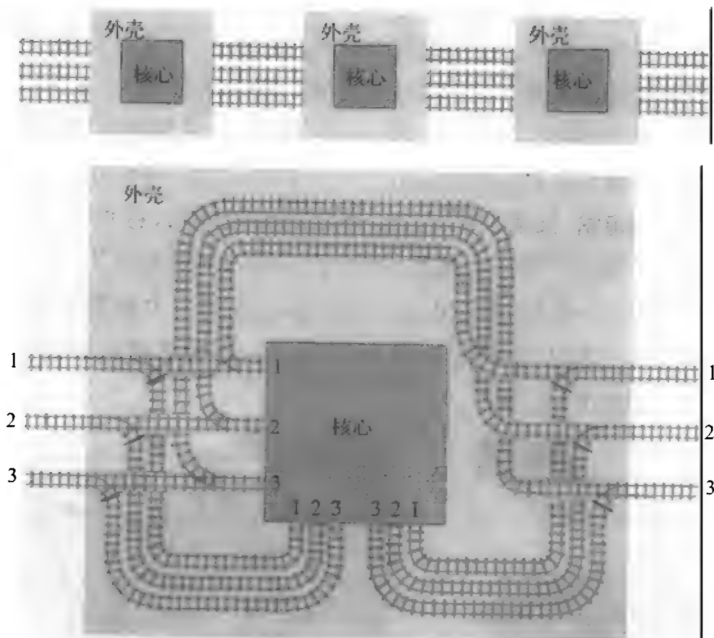


图 2　图灵机的纸带可用一系列相同的框来代替(上图)。每个框由一个外壳和一个核心组成(下图)。外壳的作用是使任一方向进入框的列车都得到同样的处理,而核心则负责执行图灵机的规则

"是的",Dee 表示同意。"但是你在框里放些什么东西呢?薛定谔的猫?"

"我要说明一下框是如何分阶段设计出来的。列车轨道同时用输入线和输出线,因此框不会'记住'列车是从哪个方向进入的。这样框就可以建造成一个根据图灵程序使列车从两条输入线以同样方式进入核心并引导它们又开出来的外壳。然后我们就可以略去这外壳而把注意力放在核心的设计上。"

"你将需要——"Dum 起了个头。

“子程序”，Dee 接着说道。像往常一样，他们总是想在前头。子程序是程序的一部分，它可由程序的其他任何一部分反复地“调用”。把许多子程序联接在一起，就可以构造出复杂的程序。

“是的”，我表示同意。“你可以把一独立的子系统连接到整整一系列懒惰岔道口上，从而构造出一个子程序，然后列车开进来，并在进入时扳动道岔，接着它在这个子系统内到处移动，直到执行完这个子系统所计算的子程序。最后，由于它在进入时的扳道方式，它将沿着进入时的同一条轨道驶出去。每条输入线用一个懒惰岔道口，就可以使所有列车从左方进入，执行子程序后再沿着进入时的同一条轨道向右方驶出(见图 3)。”

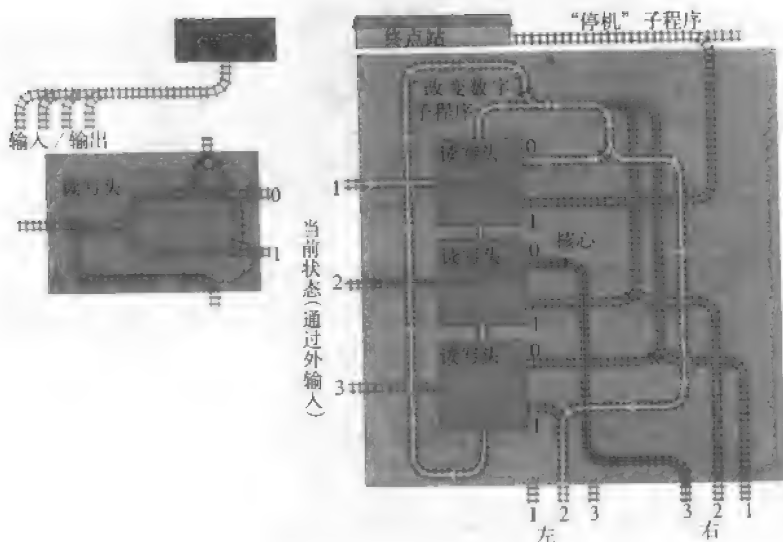


图 3　在核心部分(右)，列车从左边进入，服从相应的图灵机规则，然后从底部离开。上图所示为一个子程序的布局：列车通过懒惰岔道口进入，并沿同一条轨道离开。子程序下方是一个读写头；注意有一个触发器岔路口



"嗯,正确。"

"好了,你现在还需要一件东西:读写头。如果列车从左方进入一个读写头,则它将根据纸带上的'当前'点的数字沿线路 0 或线路 1 驶出。如果列车从上方进入,则它将交换 0 和 1 并从下方驶出。为此,懒惰岔道口 P 的扳道位置要能根据'纸带'在该方格上的数字重新引导列车沿输出线 0 或 1 驶出。触发器岔道口的置定,应使得从上方进入的第一列列车把 P 扳到另一个位置上。"

"有了所有这些组成部分,就可以构造框的内核了(见图 3 的右图)。你可能需要一些桥,以避免轨道的交叉,不过我们可忽略这个问题。该核心由一组并列的读写头构成,框的每一内部状态有一个读写头。输出线 0 和 1 通到核心的输出线中的一条上,或者通到把列车引入改变纸带上该方格的状态的子程序的懒惰岔道口上,或者通过一个把列车引入单个终点站的'停机'子程序上。"

"让我看看",Dee 插嘴说,"对于你的这个例子,有一条规则'状态 1,数字 0:改变数字,向左移一格,进入状态 2。'这怎样做到?"

"处于状态 1 意味着列车从侧面沿线路 1 进入方格。这一状态实际上是由先前方格的输出置定的,该输出引导列车在离开时进入线路 1。在这种情况下,'写'在当前方格上的数字是 0:也就是说,读写头中的所有懒惰岔道口都置定在 0 上。这样列车就进入第一读写头,沿线路 0 离开并驶入一组缓冲岔道口。这些岔道口引导列车进入'改变数字'子程序。列车竖直向下行驶,穿过所有的读写头,并把它们的状态从 0 改变为 1。因此写在当前方格中的数字现在就是 1 而不是 0。列车继续沿读写头左边的垂直轨道回驶,驶出子程序而回到其原先的轨道上,然后沿左边的输出线 2 离开核心,这样实际上就如所要求的那样使列车进入左边的方格,且处于状态 2 中。"

"妙极了。假定我们考虑这一条规则。'状态 2,数字 0:让数字保持不变,向左边移一格,进入状态 3。列车沿线路 2 进入,并沿线



路0离开读写头,而线路0直接通向右边的出口线路3。由于列车不会靠近子程序循环,因此方格的状态保持不变。"

"不错",Dum说道,"同样明显的是,规则'状态1,数字1:停机'也能完满地实现。列车沿线路1进入,并沿线路1离开读写头,这样列车就被直接送到了通向终点站的线路上。"

"是的。你完全按定义图灵机的规则铺设了这些线路。"

"你是否意识到",Dee问,"这表明一个列车系统的未来行为可以是不可判定的?"

"当然",Dum说,"图灵证明了图灵机的停机问题在形式上是不可判定的。你可以建造一台图灵机,对它来说没有任何办法能事先判定计算是否会终止。"

"这意味着对相应的列车系统来说,你无法事先预测列车是不是会到达终点。"

"这太令人吃惊了",我指出,"我从不去为理论数学问题的形式不可判定性大伤脑筋。我的意思是,谁去操那分心呢?但下面这种情况却有点让人感到不安了:你可以建立一个机械系统——玩具列车轨道,看在老天份上——它的运行原理是完全透明的,但却不能回答诸如该列车是否会抵达一个选定的车站这样简单的问题。"

"说起哪个——"Dee插话道。

"从上一次停车到现在已经过了好久了",Dum说。

我擦去了玻璃窗上凝结的一层水汽。"嗯",我说,"列车已经慢得好像在爬行了。我能看见的只是一个大方框,上面画了一个数字'1'。我发誓,地铁隧道里有一个标志,就是'触发器岔道口7743A/91。"

"如果地铁车辆停在两站之间,Dee的幽闭恐怖症会发作的。"

"地铁网络已经增加了一些新的连接线路",我推测道,"或许它的连通性已经超过了图灵阈限,因而它已达到了人工智能的水平。"



“呵，万能的地铁”，Dee 朗诵道，她的音调迅速地升高，“我们这些凡人祈望无所不知的你帮助我们摆脱——”正说到这里，连接门打开了，一个身穿制服的保安人员走进了车厢。

“诸位，线路上出了个小问题”，他微笑着说，“用不着担心，只是列车将要放慢行驶一会儿。”Dee 松了口气。“喂，那位小女士有什么问题吗？”

“没有”，Dum 回答说，“他刚才以为她困在一台有人工智能的图灵机里了。”

“这不是游览机(touring machine)”，那保安人员愤愤地说，“这是一辆短途旅客列车，亲爱的。”

至少我认为那就是他说的话。

46. 看似无尽，实则有穷

假如你有个篮子盛着 100 个鸡蛋。另外还有许多装鸡蛋的纸板箱。你的任务就是把所有鸡蛋装在这些纸板箱里。每一次(或者每一步)或是把一个鸡蛋放在某一个纸板箱里,或是从某个纸板箱中取出一个鸡蛋然后把它放回篮子里。你的程序要这样安排:在接连两次把鸡蛋放进纸板箱之后,就必须从纸板箱中取出一个鸡蛋放回篮子里。虽说很明显这是一种效率很低的包装鸡蛋的办法,可是到最后所有鸡蛋显然都能装进箱子里去。

现在假定篮子可以盛任意多数目有限的鸡蛋。如果你一开始随心所欲要许许多多鸡蛋,那这个任务可就大得不得了。不过,一旦最初鸡蛋的数目确定下来,要完成这个任务所需的步数就具有一个有限的上界。

如果规则准许你在任何时候,都可以把任何数目的鸡蛋再放回篮内,情况就会有根本的变化。这时完成这个任务所需的步数就不再有一个上界,甚至一开始篮子中只有两个鸡蛋时也是如此。所以,把有限数目的鸡蛋进行装箱的任务按照规则的不同,可以有个完,也可以没完没了,也可以由你挑选一个步骤使这个任务在有限步内完



成或无限地进行下去。

现在我们来考虑几个有趣的数学游戏,它们具有下面的特点。虽然实际上没有办法避免在有限多步之内完成它,但是从直观上看来你却仿佛能够把完成任务永远地拖延下去。

头一个例子是从哲学家兼作家兼逻辑学家 Raymond M. Smullyan 的一篇文章中引来的。想象你有无穷多个落袋台球,每个台球上都标记上一个正整数,而且对于每一整数,都有无穷多个台球以这个整数为其标记。你还有一个箱子,其中包含有限多个标数码的台球。你的目标是把箱子弄空。每一步要求你从箱子拿出一个台球。同时换上任意有限多个数码比它小的台球。1 号台球是唯一的例外。因为没有台球的数码比 1 还小,所以对每个 1 号台球就没有台球来替换了。

不难用有限多步就把箱子弄空。这只要把每个数码比 1 大的台球用一个 1 号台球来替换,一直到箱子里只剩下 1 号台球。然后每一次取出一个 1 号台球就行了。可是规则准许你用任意有限数目的数码较小的台球来替换一个数码大于 1 的台球。例如,你可以取出一个数码为 1 000 的台球,而换上 10 亿个数码为 999 的台球再加上 100 亿个数码为 998 的台球再加上 100 亿亿个数码 997 的台球再加上……这样一来,箱子里台球的数目在每一步的增加都大得出奇。你是否能够永远拖延下去不使箱子弄空呢? 实际上没有办法避免完成这个任务,虽然乍看起来这好像是令人不能置信的。

注意,比起鸡蛋游戏来,弄空箱子所需的步数数目要更加大得多。不仅是开始时的鸡蛋数目没有限制,而且每次你取出一个数码大于 1 的台球之后,用来替换它的台球的数目也没有限制。借用 John Horton Conway 的话来说,这个步骤是"无界的无界"。在这个游戏的每一个阶段,只要箱中还包含哪怕是一个数码大于 1 的台球,那就不可能预见要把箱子中 1 号台球之外的台球全取出需要多少步



(假如所有的台球的数码都是1,使箱子变空的步数当然就和1号台球的数目一样多)。不过,无论你替换台球的方法多巧妙,在有限多步之后箱子终究要变空的。当然,我们必须假设,尽管你不一定要长生不死,也需要活得足够长才能完成这个任务。

Smullyan 把这个惊人的结果发表在一篇论文"树与台球游戏"上。其中给出几个证明,有一个是用归纳法来简单论证的。Smullyan 的论述是如此之好,我只能原原本本引述如下:

"假如箱子里的台球全是数码1,那么显然我们输定了。假设箱子里球的数码最大是2,那在一开始我们有有限多个2号台球和有限多个1号台球。我们不能一直老把1号台球扔出去,因此迟早我们总要把其中一个2号台球拿走。这样箱子里的2号台球就少一个(不过可能却包含比我们开始时要多得多的1号台球)。现在我们还是不能老把1号台球扔出去,因此迟早我们总要扔掉另外一个2号台球。可以看出经过有限多步之后,我们必定得扔出去最后一个2号台球,这时我们又回到只有1号台球的情形。我们已经知道这种情形肯定是失败的。这就证明,当台球的最大数码是2时,这个过程必将中止。那么最大数码是3时又如何呢?我们不能一直不断地把数码小于2的台球扔出去(我们刚刚证明这点!);因此我们迟早要扔出去一个3号台球。所以,到头来我们必定要扔出去最后一个3号台球。于是这就把问题归结为上面的最大数码为2的情形,而这种情形我们已经解决了。"

Smullyan 还以树图作为这个游戏的模型来证明它必定终止。所谓"树"就是一组线段,每条线段联结两个点,而且每一个点都通过唯一的一串线段连结到某一点,这点称为树的根。一个台球游戏的头一步(即箱子满时)是把每个台球表示为一点,标上和台球一样的数码,并且用一根线把它同根连结在一起。当一个台球用数码较低的其他台球来替换时,就把其数码擦去,而替换它的那些新台球用高一



级的带数码的点来表示,并且这些点都同表示移去的台球和点连结起来。这样一来,树就稳步向上长,其"端点"(这些点不是根而且只同一线段相连接)总表示在该阶段箱子里的台球。

Smullyan 证明,如果这树能长大到无穷(即有无穷多个点),它就必定至少有一个无穷支叉永远向上延伸。不过,这显然不可能,因为沿着任何支叉的数码总是逐步减小,因此最后中止到 1。因为树是有限的,它所表示的游戏必定有个终止。正如台球的情形一样,完成这个树需要多少步也是没法在事先就预见到的。在完成阶段,树的生长就到达了其上限,所有端点所标的数码都是 1。当然,这些 1 号点的数目可能超过全宇宙电子的数目或者任何更大的数。然而,这个游戏并非西西弗斯式的,它肯定在有限步之后终止。

Smullyan 首先以台球游戏为模型来表示的基本定理是由有关集合顺序的定理推出来的,而这些定理可追溯到 Georg Cantor 关于超限序数的工作。它和有限树的无穷集的一个深刻的定理密切相关,这个定理首先由 Joseph B. Kruskal 证明,后来 C. St. J. A. Nash-Williams 把证明简化。最近 Nachum Dershowitz 和 Zohar Manna 用同样的论证方法证明某些涉及"无界的无界"运算的计算机程序必定最后导致停机。

Smullyan 台球游戏的一个特殊情形在图 1 的左图中用一棵有限树表示,这棵树从根向上依次标记了数码。它准许我们砍掉任何端点,连同其上连接的线段,然后在这树上随便加上多少枝叉,加在哪里都可以,只要所有的新点比你去掉的点的数码小就行。例如,图 1 中右图表示在一个 4 号点被砍去之后,树的生长的一种可能情形。尽管砍掉一个点之后,树上可能长出几亿几亿个新枝叉,但是经过有限多次砍去之后,整个树都将被砍倒。与更一般的台球游戏不一样,这里我们不能去掉任何我们想去的点,而只能去掉端点,但是因为每个去掉的点都是用数码更小的点来替换,所以 Smullyam 的台球定理



仍然适用。在每次砍伐后，树可能长得更加繁茂，但是从某种意义上来讲，它总是越来越接近地面，直至最后完全消失。

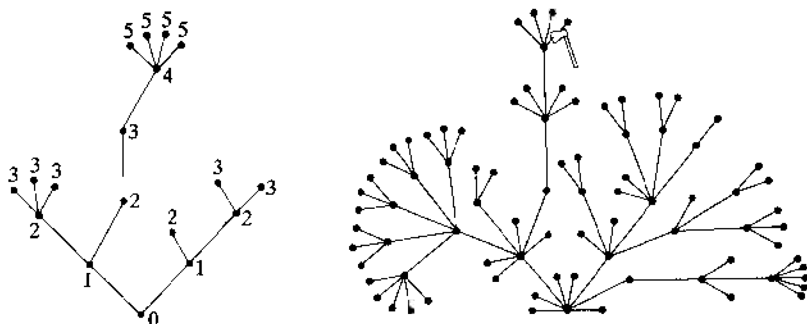


图1　修剪树的问题

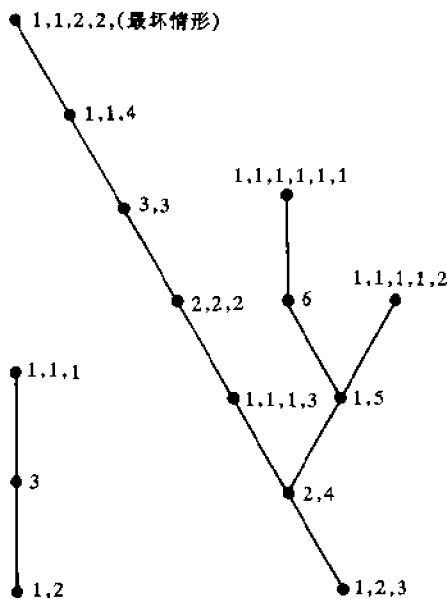


图2　三张牌和六张牌的独玩游戏中的树



关于这种看起来仿佛可以永远继续下去而实际上办不到的问题，我们可以再举一个例子，它以18点问题著称。你从一条线段开始。在其上你任选一点。现在选第二个点使得这两个点处于该线段不同的一半之内（两个一半都取为“闭区间”，也就是说端点并不看做在区间之“内”）。再放置第三个点，使这三个点的第一个都在线段的不同1/3区间之内。现在很明显，前面两点根本不可能任意放置。比如说，它们不能都放在线段的中间彼此靠得很近的地方，也不能都在一个端点处靠得很近。必须仔细地放置它们，以使第三个点加进去时三个点中的每一个都在线段不同的1/3一区间之内。按照这种方式进行下去，当放置第n个点时使得前n个点总是占据线段不同的$\frac{1}{n}$部分。假如你仔细地选择位置，你在这条线段上最多能放置多少点？

直观上看，好像能放置无穷多点，一个线段显然可以分成随意多少等分，其中每等分可以包含一个点。关键在于这些点必须顺序地编号以满足问题的条件。令人吃惊的是，结果你根本不能这样放置17个以上的点！不论你安置17个点多么巧妙，第18个点肯定会破坏我们的规定，从而游戏告终。事实上，放置10个点也不是一件容易的事。

这个不一般的问题首先由波兰数学家Hugo Steinhaus发表在《初等数学一百题》中（问题6和7）。Steinhaus给出一个14点的解答，他在一个脚注中提到M. Warmus已经证明17个点是个极限情形。第一个发表的证明是Elwyn R. Berlekamp和Ronald L. Graham在他们的论文“有限序列的分布的不规则性”中给出的。

华沙数学家Warmus一直到6年之后才在同一杂志上发表他那较为简短的证明。他给出一个17点的解答，并补充说17点解有768种图案；要是你把它们的倒头图案看成是不一样的，那就有1 536种。

图 3　10 张牌的保加利亚树

关于这种与直观相反地突然终止的难题，我们还可以举最后一个例子。你可以用一副扑克牌来做它的模型。它的来源不太清楚，但是，把这个难题告诉我的 Graham 说，欧洲数学家把它称为保加利亚的单人独玩牌戏，这样叫的理由他一直也没能弄清楚。级数 1+2+3……的部分和称为三角形数，因为它们对应于三角形阵列，比如滚木球游戏中的 10 个木瓶以及落袋台球游戏的 15 个球台。玩保加利亚游戏需要用任一三角形数那么多张扑克牌。从一副标准扑克牌



中你能得到的最大的三角形数是45，它是前9个正整数之和。

把45张牌放成一迭，然后把它们随便分成多少迭，每一迭可以有随便多少张。你可以使它仍然保持成一迭，也可以把它分成两迭、三迭或多迭，从哪里分都可以，甚至可以做44次分牌，分成45迭，每迭只有一张牌。现在开始重复进行下面的步骤。从每一迭中各取出一张牌，把这些取出的牌都放在桌上并成一新迭。这些迭不一定非得排成一行。你可以随便放置。然后重复这个步骤做成另外一迭，并不停地这样做下去。

由于各迭的结构一直以不规则的方式不断地变化，因而似乎不太可能到达这样一种状态，即只有一迭有一张牌，一迭有两张牌，一迭有三张牌，如此下去，直到最后一迭有九张牌。要是你能达到这种不太可能的状态而不陷进循环往复（即游戏总会回复到先前的某一状态），那游戏就必定终止，因为此时状态已经不能变化。重复上述步骤总使牌处于同以前完全相同的相继整数状态。令人惊奇的是，不管游戏初始状态为何，结果你总会在有限步之内达到这个相继整数状态。

保加利亚的单人独玩牌戏是分拆理论中某些问题的一种模型，它们决不是很简单的。一个正整数n的分拆就是一个正整数能够表示为正整数之和的全部方式而不论这些和数的顺序。例如，三角形数3有三种分拆：1+2、1+1+1，3。当你把一堆纸牌分成任意多迭叠、每迭任意多张时，你就给这堆纸牌建立了一个分拆。保加利亚单人独玩牌戏就是把一种分拆变成另一种分拆的方法，即把每个和数减1，然后把被减的1的总数目作为分拆中的一个新数。这个游戏总是得出一串不相重复的分拆，其中最后一个是相继整数的分拆。这个事实决不是显而易见的。据说它是由丹麦数学家Jφrgen Brandt在1981年首先证明的。但是我并不知道他的证明内容，也不知道它是否已经发表。

对于任何三角形数的牌数，保加利亚单人独玩牌戏可以用图表示为树，其根用相继整数分拆来标记，而其他的分拆用树的点来表示。图4中左图表示三张牌的游戏的单树、右图则表示六张牌的11种分拆的树，它比较复杂。任何游戏都以相继整数分拆告终这个定理等价于这样一个定理，即所有三角形数的分拆可以用连通的树来表示，其中每一个分拆比它在游戏中的后继者高一级，而树的根就是相继整数分拆。

注意在六张牌的树中，从最高点到根部共有六级。最高点的分拆，即1,1,2,2，是"最坏的"初始情形。不难看出，从任何分拆出发，游戏必定在六步之内终止。有人猜想，如某一游戏中的牌数为$\frac{1}{2}k(k+1)$，则该游戏必定在$k(k-1)$步之内终止。去年计算机科学家Donald E. Knuth要他在斯坦福大学的学生用计算机来验证这个猜想。对于k小于或等于10，他们证实了这个猜想，因此，这个猜想几乎肯定是对的，但是至今证明还没能得到。

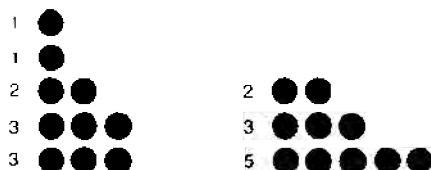


图4　共轭对1,1,2,3,3和2,3,5

图3表示保加利亚单人牌戏在10张牌(k=4)情形下的树。现在在顶上有三个最坏的情形，每一个离根都有12级。还要注意有14个端点，我们称之为Eden(伊甸园)分拆，因为除非你从它们出发，否则它们是不会在游戏中出现的。它们都是这样分拆，其分拆的部分数目比部分中最大数目大2或更多。

图4中左图是用点来表示树顶上的分拆1,1,2,3,3的标准方



法。如果把这个图案进行转动并作镜面反射,它就变成图4中右图的图案。它的三行现在给出分拆2,3,5。这两个分拆中的每一个都称为另一个的共轭。共轭关系显然是对称的。如一分拆的共轭仍为其自身,则称为自共轭分拆。在10的分拆的树上只有两个自共轭的分拆,一个是根,一个是1,1,1,2,5。如果把其余的分拆按共轭来配对的话,沿着树干就会出现一个惊人的图案。共轭分拆对正如图中字母所示,对于至今研究过的所有保加利亚树,这沿着主干的对称性都存在。

假如对于所有的树,这种对称性都存在的话,那我们就有一个简单方法来决定位于顶端的最坏情形。这个最坏情形就是正好在根上端的分拆(总是唯一一个)的共轭分拆。找出树干顶端的一个更加快的方法是把根前面附上1再从最后的数中减去1。

保加利亚游戏的操作可以用图表示。把它的一头齐的点表示图案的最左一列去掉,把这列转动90度,然后作为新行加进来。只有形如1,2,3,4……的分拆的图在这个操作下不变。假如你能够证明,除了相继整数分拆之外,没有任何分拆上的操作序列能够使一个图回到其原来状态,那你就已经证明,所有保加利亚游戏都可图示为树,所以一达到它们的根,游戏也就终止。

如果用55(k=10)张牌来玩这个游戏,则一共有451 276种分拆方式,因此画出树会十分困难。甚至于15张牌的树(共有176点),也需要计算机来帮助画图。这些分拆数是怎样算出来的呢?这里可有一段漫长而有趣的故事。让我们假定分拆是有序的,则3就会有4种有序的分拆(通常称为"合成"):1+2,2+1,1+1+1,3。结果合成的总数的公式很简单,是2^{n-1}。但如果分拆是非有序的(如在单人游戏中那样),情况就变得难以置信地杂乱无章。虽然有许多递归公式来计算分析数的阶,也就是在每一步用到所有较小的数的已知分拆数,但一直到近代才得出一个精确的渐近公式。英国数学家G.

H. Hardy 同他的印度朋友 Srinivasa Ramanujan 合作，取得了一项巨大的突破。他们的还不十分精确的公式在 1937 年由 Hans A. Rademacher 进一步加以完善。这个 Hardy-Ramanujan Rademacher 公式是一个极为冗杂的无穷级数，其中包括 π 以及双曲函数的平方根、复根以及导数！George E. Andrews 在他写的关于分拆的标准教材中，称之为“难以相信的恒等式”并说在分拆论的历史上这是“一项登峰造极的成就”。

对于 $n=1, n=2, n=3, n=4, n=5$ 和 $n=6$，分拆数序列是 1，2，3，5，7，11，因此你也许推测下一个分拆数是下一个素数，即 13。哎呀，其实它是 15。也许所有分拆数都是奇数。不对，下一个分拆数是 22。在分拆理论中，一个尚未解决的深刻问题是，随着 n 的增加，偶分拆数与奇分拆数是否渐近相等。

假如你把分拆理论仅仅看成一种数学游戏，那让我在结尾时讲一下，利用所谓 Young 氏表的数的阵列来图示分拆集合的方法，已经在粒子物理学中得到充分应用。不过这又是另外一种台球游戏了。

47. 素数王国大猎捕

大数在密码学这一领域里令人意外地大出风头，现今这一领域的中心问题就是素数和因子。例如，RSA 密码系统就是以两个大素数——比如说每个数各有 100 位左右——乘起来所得的数为基础的。本文将不对这个密码系统本身作介绍，而只是想说明两个基本的数学问题。即：

1. 素性检验问题：给定一个大数，如何确定它是不是素数？

2. 因子分解问题：给定一个大数，如何求出它的各个因子？

很明显，因子分解问题的任何一个答案都同时也解决了素性检验问题，但是因子分解问题的难度看来要大得多。截至目前，要检验一个任意的 200 位数是否素数是办得到的，但要找出它的因子则不大可能了（除非该数有一些不同寻常的特征）。这就使得探索素数王国（即所有素数的集合）比探索因子王国（所有整数构成的集合）要容易得多。但是，数论中搜寻大猎物的人已经广泛地打入了这两个领域。本文将主要介绍素性检验，以后再谈因子分解的问题。

在学校里我们已经学过一种十分简单明了的寻找因子的方法（或算法）：试除法。给定一个数（如 1997），用不超过它的平方根的



所有可能除数依次去作试除。(在这个例子里,1997 的平方根为 44 多一点。)用所有素数除数来试除就够了,但一般说来这就需要有一张素数表,因此我们通常采用一种折衷办法:用 2 以及不超过该平方根的所有奇数来进行试除。在此例里,我们将用 2、3、5……43 来依次试除——总共是 22 个除数。这 22 个数中没有一个能整除 1997,因而 1997 是一个素数。对 1921 这个数用同样方法作试除,我们发现它有一个因子:17。然后我们算出 1921÷17=113,并对 113 这个数再次使用试除法,发现它是个素数,因此 1921 的最终因子分解结果是:1921=17×113。

对于较小的数来说,这一方法相当管用,但对于因子王国的遥远角落来说,这方法就完全失效了。对于 98765432123456789 这样的数(它只有 17 位),试除法所需的除数大约有 1.57 亿万个。一般说来,对于数 n,此方法需要进行大约$\sqrt{n}/2$ 次试除。如果 n 为 100 位数,那么试除的次数就在 10^{50}这一数量级上。用速度最快的超级计算机来进行试除,即使运行了相当于现今宇宙寿命那样长的时间,也不过就是刚刚起步而已。

估计算法所需的运行时间属于一个比较新的数学分支(称为复杂性理论)的研究范围。这类估计主要是针对十进制的数,它们发现$\sqrt{n}/2$ 和$\sqrt{n}$的复杂性没有实质的差别。事实上,$\sqrt{n}$的位数或者和$\sqrt{n}/2$相同,或者只多 1 位。你可以称这种论证为"大数不变性原理"(Principle of Permanence of Large Numbers)。在评估各种方法的有效性时,可以将通过这种方法简化表达式,只要位数变化不大的话。

欧几里得证明了素数有无穷多个,因此是永远找不完的。但素数的分布却是越大越稀疏。Carl Friedrich Gauss 猜测小于 n 的素数的个数逐渐趋近于 n/logn(此式中的对数是"自然"对数,其底为 e=2.71828……),后来 Jacgues Hadamard 和 Charles de la Vallee Poussin 证明了这一猜想。素数的分布看起来是随机的。它不可能是真正随



机的，但的确看起来无法预测。

有多种方法可在不寻找一个数的因子的情况下证明该数是素数。一个例子是1770年提出的Wilson定理：当且仅当n整除$(n-1)!+1$时，n为素数。这里$m!=1\times2\times3\times\cdots\cdots\times m$，即阶乘函数。例如，$10!+1=3\ 628\ 801$，它可被11整除，因此11是素数。遗憾的是，从复杂性分析的角度来看，这一方法比试除法还要糟，因为它需要进行大约n次乘法(而试除法只要进行$\sqrt{n}$次除法)。

现代的素性检验方法基于一个其起源可追溯到古代中国的类似发现：如果n为素数，则它可整除2^n-2。看来中国人认为这一定理的逆定理也是正确的；如果真是如此，那就可以运用他们的这一准则作为一种素性检验方法。但是，计算2^n-2并检验它是否能被n整除是不是也需要n次左右的乘法呢？奇妙得很，答案是否定的，而我们将以此为出发点开始认真探索大数不变性原理。

出于论证的需要，想象一下我们对107这个数是否素数感兴趣，并打算检查它是否能整除$2^{107}-2$。看起来我们似乎需要106次乘法：$2,2\times2,2\times2\times2\cdots\cdots$但是还有另外一种方法。首先，我们只计算$2^2,2^4,2^8$等等，在这个例子中就是一直算到$2^{64}$，注意这些乘方中的指数都是2的幂。我们通过反复求平方来进行上述计算：$2^2=2\times2$，$2^4=2^2\times2^2,2^8=2^4\times2^4\cdots\cdots2^{64}=2^{32}\times2^{32}$。这一计算总共只要进行6次乘法。

现在我们把107写成二进制形式，即$107=1\ 101\ 011$。这就意味着$107=1\times64+1\times32+0\times16+1\times8+0\times4+1\times2+1\times1$，由此可推出$2^{107}=2^{64}\times2^{32}\times2^8\times2^2\times2^1$，只要在算出$2^{64}$的基础上再作四次乘法就行了。简而言之，我们只需要作10次乘法(而不是106次)就可以求出2^{107}。

数字越大，节省的计算次数就越多。复杂性分析证明，用这一方法计算2^n的值至多需要$2\log_2 n+1$次乘法。这里$\log_2 n$表示以2为



底 n 的对数，它大致等于 n 的十进制形式的位数的 3.4 倍。因此，当 n 有 100 个十进制数位时，计算 2^n 只需要作 340 次乘法，而不是通过反复乘以 2 来计算 2^n 时所需要的 10^{100} 次左右的乘法。

读者可能担心，计算中涉及的数字将变得相当大，这样乘法步骤将越来越长。事实上，我们不需要完全计算出 2^n-2，我们的目标只是想知道它能不能被 n 整除。奥妙在于，所有这些算术运算都是以 n 为模进行的——这就意味着我们可以忽略 n 的任何整数倍数。在计算的任何一步上，所有数字均可以用它们在除以 n 后所得的余数来代替。

对于很大的 n，这一简化将起巨大的作用。假定 n 有 101 位数，那么 2^n-2 的"初始"值将有大约 $10^{99.5}$ 位数。注意，2^n-2 的位数不是 99.5，而是 10 的 99.5 次方。这个数字大得连整个宇宙都写不下：即使我们能够在现有的每个电子上写一位数，宇宙中的电子总数也不够用。相比之下，进行模 10^{100} 的算术运算意味着任何一个数字都不超出 100 位。

虽然中国人的上述猜想不能作为可靠的素性检验方法，但它的例外情况——即 n 非素数但却能整除 2^n-2，此时 n 又称为以 2 为底的伪素数——却是很罕见的（伪素数的一个例子是 $341=11\times31$）。许多这类例外情况可以用中国检验法的一种推广形式来处理，此推广是费马（Pierre de Fermat）所证明的，称为费马小定理。该定理是说，如果 n 为素数，则对于任何一个 a，n 均可整除 a^n-a。这样，我们可以不用 2^n-2 进行检验，而用 3^n-3、5^n-5 等等来进行检验。不过对此仍然存在一些例外情况，即有的数能够通过新的可除性检验但却不是素数。这些数称为以 a 为底的伪素数。有的数可能同时是几个不同的底的伪素数。例如 $2701=37\times73$ 就是以 2 和 3 为底的伪素数。但是一个数是许多不同的底的伪素数这种情况是极为罕见的。小于 250 亿的数中，唯一的一个同时以 2，3，5 和 7 为底的伪素



数是 3215031751——但它不是以 11 为底的伪素数。这样,在知道了这个唯一的例外的情况下,把不多的几种中国式检验法结合起来,我们就可以迅速地检验小于 250 亿的任何一个数是不是素数。

令人遗憾的是,每一个底都有无穷多个相应的伪素数,因此我们无法找到一个真正合适的底并只用这个底来检验素数。事实上,情况反而更糟:存在着一类所谓 Carmichael 数,它们是以任何一个与该数本身没有公共因子的数为底的伪素数。最小的 Carmichael 数是 $561=3\times11\times17$;其后依次为 1105、1729、2465 和 2821。1912 年,Robert D. Carmichael 发现了 15 个这样的数,并提出"这一清单可以无限地延伸下去"。他的猜想在 1992 年得到最终证明:比 n 小的 Carmichael 数至少有 $n^{2/7}$ 个。

为了避开伪素数这一麻烦问题,Gary L. Miller 发现了费马检验法的一个更复杂的变种,并把这一新检验法的例外情况称为强伪素数。每一个非素数都至少有一个"证人",也就是以其为底时它不是强伪素数的数。如果能够证明这些证人足够小,那么就可以通过逐一尝试所有可能的证人来检验一个数是不是素数。借助于数论中一个著名的未解决问题(即 Riemann 假设),Miller 证明了每一个非素数 n 均有一个其大小不超过 $70(\mathrm{Log}n)^2$ 的证人。由于 Logn 比 n 小得多,因此这一估计值大大改进了素性检验的效率。

1980 年,Adleman 和 Robert S. Rumely 发现了一种方法来改进 Miller 检验,使其可以不依赖 Riemann 假设。这种方法的运行时间要稍微长一些,但完全适用于解决比如说 2 百位数的素性检验问题。随后还研究出了其他一些更高级的素性检验方法,特别引人注意的是 Hendrik W. Lenstra, Jr. 的椭圆曲线算法。

这些发现使数学家们处于一种容易失望的境地。例如,给定一个有 200 位的数,他们可以在一台通常的台式计算机上迅速进行检验,以确定该数是否是素数。如果答案是否定的,那么此数必定有因



子。然而这一方法对于它究竟有哪些因子却没有提供任何线索。要找出这些因子是非常困难的。不过,已经出现了一些巧妙的新设想,因子分解问题还是可以解决的。如果它真的得到解决,许多加密系统就将遇到麻烦。